Gleb Beliakov, Ana Pradera, Tomasa Calvo

Aggregation Functions: A Guide for Practitioners

Studies in Fuzziness and Soft Computing, Volume 221

Editor-in-chief
Prof. Janusz Kacprzyk
Systems Research Institute
Polish Academy of Sciences
ul. Newelska 6
01-447 Warsaw
Poland
E-mail: kacprzyk@ibspan.waw.pl

Further volumes of this series can be found on our homepage: springer.com

Vol. 208. Oscar Castillo, Patricia Melin, Janusz Kacprzyk, Witold Pedrycz (Eds.)
Hybrid Intelligent Systems, 2007
ISBN 978-3-540-37419-0

Vol. 209. Alexander Mehler, Reinhard Köhler
Aspects of Automatic Text Analysis, 2007
ISBN 978-3-540-37520-3

Vol. 210. Mike Nachtegael, Dietrich Van der Weken, Etienne E. Kerre, Wilfried Philips (Eds.)
Soft Computing in Image Processing, 2007
ISBN 978-3-540-38232-4

Vol. 211. Alexander Gegov
Complexity Management in Fuzzy Systems, 2007
ISBN 978-3-540-38883-8

Vol. 212. Elisabeth Rakus-Andersson
Fuzzy and Rough Techniques in Medical Diagnosis and Medication, 2007
ISBN 978-3-540-49707-3

Vol. 213. Peter Lucas, José A. Gámez, Antonio Salmerón (Eds.)
Advances in Probabilistic Graphical Models, 2007
ISBN 978-3-540-68994-2

Vol. 214. Irina Georgescu
Fuzzy Choice Functions, 2007
ISBN 978-3-540-68997-3

Vol. 215. Paul P. Wang, Da Ruan, Etienne E. Kerre (Eds.)
Fuzzy Logic, 2007
ISBN 978-3-540-71257-2

Vol. 216. Rudolf Seising
The Fuzzification of Systems, 2007
ISBN 978-3-540-71794-2

Vol. 217. Masoud Nikravesh, Janusz Kacprzyk, Lofti A. Zadeh (Eds.)
Forging New Frontiers: Fuzzy Pioneers I, 2007
ISBN 978-3-540-73181-8

Vol. 218. Masoud Nikravesh, Janusz Kacprzyk, Lofti A. Zadeh (Eds.)
Forging New Frontiers: Fuzzy Pioneers II, 2007
ISBN 978-3-540-73184-9

Vol. 219. Roland R. Yager, Liping Liu (Eds.)
Classic Works of the Dempster-Shafer Theory of Belief Functions, 2007
ISBN 978-3-540-25381-5

Vol. 220. Humberto Bustince, Francisco Herrera, Javier Montero (Eds.)
Fuzzy Sets and Their Extensions: Representation, Aggregation and Models, 2007
ISBN 978-3-540-73722-3

Vol. 221. Gleb Beliakov, Tomasa Calvo and Ana Pradera
Aggregation Functions: A Guide for Practitioners, 2007
ISBN 978-3-540-73720-9

Gleb Beliakov
Ana Pradera
Tomasa Calvo

Aggregation Functions: A Guide for Practitioners

With 181 Figures and 7 Tables

Springer

Dr. Gleb Beliakov
School of Engineering and IT
Deakin University
221 Burwood Hwy
Burwood 3125
Australia
gbeliako@gmail.com

Dr. Ana Pradera
Escuela Superior de Ciencias
Experimentales y Tecnología
Universidad Rey Juan Carlos. c/ Tulipan s/n
28933 Móstoles, Madrid
Spain
ana.pradera@urjc.es

Professor Tomasa Calvo
Escuela Técnica Superior de Ingeniería Informática
Universidad de Alcalá
Crta de Madrid-Barcelona Km 33, 6
29971Alcalá de Henares, Madrid
Spain
tomasa.calvo@uah.es

Library of Congress Control Number: 2007932777

ISSN print edition: 1434-9922
ISSN electronic edition: 1860-0808
ISBN 978-3-540-73720-9 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media
springer.com

Typesetting: Integra Software Services Pvt. Ltd., India
Cover design: WMX Design, Heidelberg

Printed on acid-free paper SPIN: 12047963 89/3180/Integra 5 4 3 2 1 0

To

Gelui Patricia, Chaquen and Sofia
G.B.

Ultano
A.P.

Raquel, Carmina, Adela and Carlos
T.C.

Preface

The target audience of this book are computer scientists, system architects, knowledge engineers and programmers, who face a problem of combining various inputs into a single output. Our intent is to provide these people with an easy-to-use guide about possible ways of aggregating input values given on a numerical scale, and ways of choosing/constructing aggregation functions for their specific applications.

A prototypical user of this guide is a software engineer who works on building an expert or decision support system, and is interested in how to combine information coming from different sources into a single numerical value, which will be used to rank the alternative decisions. The complexity of building such a system is so high, that one cannot undertake a detailed study of the relevant mathematical literature, and is rather interested in a simple off-the-shelf solution to one of the many problems in this work.

We present the material in such a way that its understanding does not require specific mathematical background. All the relevant notions are explained in the book (in the introduction or as footnotes), and we avoid referring to advanced topics (such as algebraic structures) or using pathological examples (such as discontinuous functions). While mathematically speaking these topics are important, they are well explained in a number of other publications (some are listed at the end of the introduction). Our focus is on practical applications, and our aims are conciseness, relevance and quick applicability.

We treat aggregation functions which map several inputs from the interval $[0, 1]$ to a single output in the same interval. By no means this is the only possible framework for aggregating the inputs or performing information fusion – in many cases the inputs are in fact discrete or binary. However it is often possible and useful to map them into the unit interval, for example using degrees of membership in fuzzy sets. As we shall see, even in this simplified framework, the theory of aggregation is very rich, so choosing the right operation is still a challenge.

As we mentioned, we present only the most important mathematical properties, which can be easily interpreted by the practitioners. Thus effectively

this book is an introduction to the subject. Yet we try to cover a very broad range of aggregation functions, and present some state-of-the-art techniques, typically at the end of each section.

Chapter 1 gives a broad introduction to the topic of aggregation functions. It covers important general properties and lists the most important prototypical examples: means, ordered weighted averaging (OWA) functions, Choquet integrals, triangular norms and conorms, uninorms and nullnorms. It addresses the problem of choosing the right aggregation function, and also introduces a number of basic numerical tools: methods of interpolation and smoothing, linear and nonlinear optimization, which will be used to construct aggregation functions from empirical data.

Chapters 2 – 4 give a detailed discussion of the four broad classes of aggregation functions: averaging functions (Chapter 2), conjunctive and disjunctive functions (Chapter 3) and mixed functions (Chapter 4). Each class has many distinct families, and each family is treated in a separate section. We give a formal definition, discuss important properties and their interpretation, and also present specific methods for fitting a particular family to empirically collected data. We also provide examples of computer code (in C++ language) for calculating the value of an aggregation function, various generalizations, advanced constructions and pointers to specific literature.

In Chapter 5 we discuss the general problem of fitting chosen aggregation functions to empirical data. We formulate a number of mathematical programming problems, whose solution provides the best aggregation function from a given class which fits the data. We also discuss how to evaluate suitability of such functions and measure consistency with the data.

In Chapter 6 we present a new type of interpolatory aggregation functions. These functions are constructed based on empirical data and some general mathematical properties, by using interpolation or approximation processes. The aggregation functions are general (i.e., they typically do not belong to any specific family), and are not expressed via an algebraic formula but rather a computational algorithm. While they may lack certain interpretability, they are much more flexible in modeling the desired behavior of a system, and numerically as efficient as an algebraic formula. These aggregation functions are suitable for computer applications (e.g., expert systems) where one can easily specify input–output pairs and a few generic properties (e.g., symmetry, disjunctive behavior) and let the algorithm build the aggregation functions automatically.

The final Chapter 7 outlines a few classes of aggregation functions not covered elsewhere in this book, and presents various additional properties that may be useful for specific applications. It also provides pointers to the literature where these issues are discussed in detail.

Appendix A outlines some of the methods of numerical approximation and optimization that are used in the construction of aggregation functions, and provides references to their implementation. Appendix B contains a number of problems that can be given to undergraduate and graduate students.

This book comes with a software package AOTool, which can be freely downloaded from `http://www.deakin.edu.au/~gleb/aotool.html`. AOTool implements a large number of methods for fitting aggregation functions (either general or from a specific class) to empirical data. These methods are described in the relevant sections of the book. AOTool allows the user to load empirical data (in spreadsheet format), to calculate the parameters of the best aggregation function which fits these data, and save these parameters for future use. It also allows the user to visualize some two-dimensional aggregation functions.

We reiterate that this book is oriented towards practitioners. While basic understanding of aggregation functions and their properties is required for their successful usage, the examples of computer code and the software package for building these functions from data allow the reader to implement most aggregation functions in no time. It takes the complexity of implementation off the users, and allows them to concentrate on building their specific system.

Melbourne, Móstoles, Alcalá de Henares
May 2007

Gleb Beliakov
Ana Pradera
Tomasa Calvo

Notations used in this book

$\Re$	the set of real numbers;
$\mathcal{N}$	the set $\{1, 2, \ldots, n\}$;
2^X	the power set (i.e., the set of all subsets of the set X);
$\mathcal{A}^c$	the complement of the set $\mathcal{A}$;
$\mathbf{x}$	n-dimensional real vector, usually from $[0,1]^n$;
$< \mathbf{x}, \mathbf{y} >$	scalar (or dot) product of vectors $\mathbf{x}$ and $\mathbf{y}$;
$\mathbf{x}_\searrow$	permutation of the vector $\mathbf{x}$ which arranges its components in non-increasing order;
$\mathbf{x}_\nearrow$	permutation of the vector $\mathbf{x}$ which arranges its components in non-decreasing order;
f_n	a function of n variables, usually $f_n : [0,1]^n \to [0,1]$;
F	an extended function $F : \bigcup\limits_{n \in \{1,2,\ldots\}} [0,1]^n \to [0,1]$;
$f \circ g$	the composition of functions f and g;
g^{-1}	the inverse of the function g;
$g^{(-1)}$	the pseudo-inverse of the function g;
N	a strong negation function;
v	a fuzzy measure;
$\mathbf{w}$	a weighting vector;
log	the natural logarithm;
	Averaging functions
$M(\mathbf{x})$	arithmetic mean of $\mathbf{x}$;
$M_{\mathbf{w}}(\mathbf{x})$	weighted arithmetic mean of $\mathbf{x}$ with the weighting vector $\mathbf{w}$;
$M_{\mathbf{w},[r]}(\mathbf{x})$	weighted power mean of $\mathbf{x}$ with the weighting vector $\mathbf{w}$ and exponent r;
$M_{\mathbf{w},g}(\mathbf{x})$	weighted quasi-arithmetic mean of $\mathbf{x}$ with the weighting vector $\mathbf{w}$ and generator g;
$Med(\mathbf{x})$	median of $\mathbf{x}$;
$Med_a(\mathbf{x})$	a-median of $\mathbf{x}$;
$Q(\mathbf{x})$	quadratic mean of $\mathbf{x}$;
$Q_{\mathbf{w}}(\mathbf{x})$	weighted quadratic mean of $\mathbf{x}$ with the weighting vector $\mathbf{w}$;

$H(\mathbf{x})$	harmonic mean of $\mathbf{x}$;
$H_{\mathbf{w}}(\mathbf{x})$	weighted harmonic mean of $\mathbf{x}$ with the weighting vector $\mathbf{w}$;
$G(\mathbf{x})$	geometric mean of $\mathbf{x}$;
$G_{\mathbf{w}}(\mathbf{x})$	weighted geometric mean of $\mathbf{x}$ with the weighting vector $\mathbf{w}$;
$OWA_{\mathbf{w}}(\mathbf{x})$	ordered weighted average of $\mathbf{x}$ with the weighting vector $\mathbf{w}$;
$OWQ_{\mathbf{w}}(\mathbf{x})$	quadratic ordered weighted average of $\mathbf{x}$ with the weighting vector $\mathbf{w}$;
$OWG_{\mathbf{w}}(\mathbf{x})$	geometric ordered weighted average of $\mathbf{x}$ with the weighting vector $\mathbf{w}$;
$C_v(\mathbf{x})$	discrete Choquet integral of $\mathbf{x}$ with respect to the fuzzy measure v;
$S_v(\mathbf{x})$	discrete Sugeno integral of $\mathbf{x}$ with respect to the fuzzy measure v;
	Conjunctive and disjunctive functions
T	triangular norm;
S	triangular conorm;
T_P, T_L, T_D	the basic triangular norms (product, Łukasiewicz and drastic product);
S_P, S_L, S_D	the basic triangular conorms (dual product, Łukasiewicz and drastic sum);
C	copula;
	Mixed functions
U	uninorm;
V	nullnorm;
$U_{T,S,e}$	uninorm with underlying t-norm T, t-conorm S and neutral element e;
$V_{T,S,a}$	nullnorm with underlying t-norm T, t-conorm S and absorbent element a;
$E_{\gamma,T,S}$	exponential convex T-S function with parameter γ and t-norm and t-conorm T and S;
$L_{\gamma,T,S}$	linear convex T-S function with parameter γ and t-norm and t-conorm T and S;
$Q_{\gamma,T,S,g}$	T-S function with parameter γ, t-norm and t-conorm T and S and generator g;
$O_{S,\mathbf{w}}$	S-OWA function with the weighting vector $\mathbf{w}$ and t-conorm S;
$O_{T,\mathbf{w}}$	T-OWA function with the weighting vector $\mathbf{w}$ and t-norm T;
$O_{S,T,\mathbf{w}}$	ST-OWA function with the weighting vector $\mathbf{w}$, t-conorm S and t-norm T.

Acronyms and Abbreviations

LAD	least absolute deviation
LP	linear programming
LS	least squares
OWA	ordered weighted averaging
QP	quadratic programming
t–norm	triangular norm
t–conorm	triangular conorm
s.t.	subject to
w.r.t.	with respect to
WOWA	weighted ordered weighted averaging

Contents

1

Introduction

1.1 What is an aggregation function?

A mathematical function is a rule, which takes an input — a value called argument, and produces an output — another value. Each input has a *unique* output associated to it. A function is typically denoted by $y = f(x)$, where x is the argument and y is the value. The argument x can be a vector, i.e., a tuple of size n: $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, and $x_1, x_2, \ldots, x_n$ are called components of $\mathbf{x}$.

It is important to understand that f can be represented in several ways:

a) as an algebraic formula (say, $f(\mathbf{x}) = x_1 + x_2 - x_3$),
b) as a graph of the function (e.g., as 2D, 3D plot, or contour plot),
c) verbally, as a sequence of steps (e.g., take the average of components of $\mathbf{x}$), or more formally, as an algorithm,
d) as a lookup table,
e) as a solution to some equation (algebraic, differential, or functional),
f) as a computer subroutine that returns a value y for any specified $\mathbf{x}$ (so called oracle).

Some representations are more suitable for visual or mathematical analysis, whereas for the use in a computer program all representations (except graphical) are equivalent, as they are all converted eventually to representation f). Some people mistakenly think of functions only as algebraic formulas. We will not distinguish between functions based on their representation, f can be given in any mentioned way. What is important though is that f consistently returns the same and unique value for any given $\mathbf{x}$.

Aggregation functions are functions with special properties. In this book we only consider aggregation functions that take real arguments from the closed interval $[0, 1]$ and produce a real value in $[0, 1]$.[1] This is usually denoted as $f : [0, 1]^n \to [0, 1]$ for functions that take arguments with n components.

[1] The interval $[0, 1]$ can be substituted with any interval $[a, b]$ using a simple transformation, see Section 1.3.

The purpose of aggregation functions (they are also called *aggregation operators* [2] , both terms are used interchangeably in the existing literature) is to combine inputs that are typically interpreted as degrees of membership in fuzzy sets, degrees of preference, strength of evidence, or support of a hypothesis, and so on. Consider these prototypical examples.

Example 1.1 (A multicriteria decision making problem). There are two (or more) alternatives, and n criteria to evaluate each alternative (or rather a preference for each alternative). Denote the scores (preferences) by $x_1, x_2, \dots, x_n$ and $y_1, y_2, \dots, y_n$ for the alternatives x and y respectively. The goal is to combine these scores using some aggregation function f, and to compare the values $f(x_1, x_2, \dots, x_n)$ and $f(y_1, y_2, \dots, y_n)$ to decide on the winning alternative.

Example 1.2 (Connectives in fuzzy logic). An object x has partial degrees of membership in n fuzzy sets, denoted by $\mu_1, \mu_2, \dots, \mu_n$. The goal is to obtain the overall membership value in the combined fuzzy set $\mu = f(\mu_1, \mu_2, \dots, \mu_n)$. The combination can be the set operation of union, intersection, or a more complicated (e.g., composite) operation.

Example 1.3 (A group decision making problem). There are two (or more) alternatives, and n decision makers or experts who express their evaluation of each alternative as $x_1, x_2, \dots, x_n$. The goal is to combine these evaluations using some aggregation function f, to obtain a global score $f(x_1, x_2, \dots, x_n)$ for each alternative.

Example 1.4 (A rule based system). The system contains rules of the form

$$\textbf{If } t_1 \textit{ is } A_1 \textbf{ AND } t_2 \textit{ is } A_2 \textbf{ AND } \dots t_n \textit{ is } A_n \textbf{ THEN ...}$$

$x_1, x_2, \dots, x_n$ denote the degrees of satisfaction of the rule predicates t_1 is A_1, t_2 is A_2, etc. The goal is to calculate the overall degree of satisfaction of the combined predicate of the rule antecedent $f(x_1, x_2, \dots, x_n)$. [3]

The input value 0 is interpreted as no membership, no preference, no evidence, no satisfaction, etc., and naturally, an aggregation of n 0s should yield 0. Similarly, the value 1 is interpreted as full membership (strongest preference, evidence), and an aggregation of 1s should naturally yield 1. This implies a fundamental property of aggregation functions, the preservation of the bounds

[2] In Mathematics, the term *operator* is reserved for functions $f : X \to Y$, whose domain X and co-domain Y consist of more complicated objects than sets of real numbers. Typically both X and Y are sets of functions. Differentiation and integration operators are typical examples, see, e.g., [256]. Therefore, we shall use the term *aggregation function* throughout this book.

[3] As a specific example, consider the rules in a fuzzy controller of an air conditioner: If temperature is HIGH AND humidity is MEDIUM THEN...

$$f(\underbrace{0,0,\ldots,0}_{n-times}) = 0 \quad \text{and } f(\underbrace{1,1,\ldots,1}_{n-times}) = 1. \tag{1.1}$$

The second fundamental property is the monotonicity condition. Consider aggregation of two inputs $\mathbf{x}$ and $\mathbf{y}$, such that $x_1 < y_1$ and $x_j = y_j$ for all $j = 2,\ldots,n$, e.g., $\mathbf{x} = (a,b,b,b), \mathbf{y} = (c,b,b,b), a < c$. Think of the j-th argument of f as the degree of preference with respect to the j-th criterion, and $\mathbf{x}$ and $\mathbf{y}$ as vectors representing two alternatives A and B. Thus B is preferred to A with respect to the first criterion, and we equally prefer the two alternatives with respect to all other criteria. Then it is not reasonable to prefer A to B. Of course the numbering of the criteria is not important, so monotonicity holds not only for the first but for any argument x_i.

For example, consider buying an item in a grocery store. There are two grocery shops close by (the two alternatives are whether to buy the item in one or the other shop), and the item costs less in shop A. The two criteria are the price and distance to the shop. We equally prefer the two alternatives with respect to the second criterion, and prefer shop A with respect to the price. After combining the two criteria, we prefer buying in shop A and not shop B.

The same reasoning applies in other interpretations of aggregation functions and their arguments, e.g., combination of membership values in fuzzy sets. Given that two objects have the same membership value in all but one of the fuzzy sets, and object A has greater membership in the remaining fuzzy set than object B, the overall membership of A in the combined fuzzy set is no smaller than that of B. Mathematically, (non-decreasing) monotonicity in all arguments is expressed as

$$x_i \le y_i \text{ for all } i \in \{1,\ldots,n\} \text{ implies } f(x_1,\ldots,x_n) \le f(y_1,\ldots,y_n). \tag{1.2}$$

We will frequently use vector inequality $\mathbf{x} \le \mathbf{y}$, which means that each component of $\mathbf{x}$ is no greater than the corresponding component of $\mathbf{y}$. Thus, non-decreasing monotonicity can be expressed as $\mathbf{x} \le \mathbf{y}$ implies $f(\mathbf{x}) \le f(\mathbf{y})$. Condition (1.2) is equivalent to the condition that each *univariate* function $f_{\mathbf{x}}(t) = f(\mathbf{x})$ with $t = x_i$ and the rest of the components of $\mathbf{x}$ being fixed, is monotone non-decreasing in t.

The monotonicity in all arguments and preservation of the bounds are the two fundamental properties that characterize general aggregation functions. If any of these properties fails, we cannot consider function f as an aggregation function, because it will provide inconsistent output when used, say, in a decision support system. All the other properties discussed in this book define specific classes of aggregation functions. We reiterate that an aggregation function can be given in any form a)–f) on p.1.

Definition 1.5 (Aggregation function). *An aggregation function is a function of $n > 1$ arguments that maps the (n-dimensional) unit cube onto the unit interval $f : [0,1]^n \to [0,1]$, with the properties*

(i) $f(\underbrace{0, 0, \ldots, 0}_{n-times}) = 0$ *and* $f(\underbrace{1, 1, \ldots, 1}_{n-times}) = 1$.

(ii) $\mathbf{x} \leq \mathbf{y}$ *implies* $f(\mathbf{x}) \leq f(\mathbf{y})$ *for all* $\mathbf{x}, \mathbf{y} \in [0,1]^n$.

It is often the case that aggregation of inputs of various sizes has to be considered in the same framework. In some applications, input vectors may have a varying number of components (for instance, some values can be missing). In theoretical studies, it is also often appropriate to consider a family of functions of $n = 2, 3, \ldots$ arguments with the same underlying property. The following mathematical construction of an *extended aggregation function* [174] allows one to define and work with such families of functions of any number of arguments.

Definition 1.6 (Extended aggregation function). *An extended aggregation function is a mapping*

$$F : \bigcup_{n \in \{1,2,\ldots\}} [0,1]^n \to [0,1],$$

such that the restriction of this mapping to the domain $[0,1]^n$ for a fixed n is an n-ary aggregation function f, with the convention $F(x) = x$ for $n = 1$.

Thus, in simpler terms, an extended aggregation function [4] is a family of $2-, 3-, \ldots$ variate aggregation functions, with the convention $F(x) = x$ for the special case $n = 1$. We shall use the notation f_n when we want to emphasize that an aggregation function has n arguments. In general, two members of such a family for distinct input sizes m and n need not be related. However, we shall see that in the most interesting cases they are related, and sometimes can be computed using one generic formula.

In the next section we study some generic properties of aggregation functions and extended aggregation functions. Generally a given property holds for an extended aggregation function F if and only if it holds for every member of the family f_n.

[4] Sometimes extended aggregation functions are also referred to as aggregation operators, see footnote 2.

Examples

$$\begin{array}{cl}
\text{Arithmetic mean} & f_n(\mathbf{x}) = \dfrac{1}{n}(x_1 + x_2 + \ldots + x_n).\\
\text{Geometric mean} & f_n(\mathbf{x}) = \sqrt[n]{x_1 x_2 \ldots x_n}.\\
\text{Harmonic mean} & f_n(\mathbf{x}) = \dfrac{n}{\frac{1}{x_1} + \frac{1}{x_2} + \ldots + \frac{1}{x_n}}.\\
\text{Minimum} & \min(\mathbf{x}) = \min\{x_1, \ldots, x_n\}.\\
\text{Maximum} & \max(\mathbf{x}) = \max\{x_1, \ldots, x_n\}.\\
\text{Product} & f_n(\mathbf{x}) = x_1 x_2 \ldots x_n = \displaystyle\prod_{i=1}^{n} x_i.\\
\text{Bounded sum} & f_n(\mathbf{x}) = \min\{1, \displaystyle\sum_{i=1}^{n} x_i\}.
\end{array}$$

Note that in all mentioned examples we have extended aggregation functions, since the above generic formulae are valid for any $n > 1$.

1.2 What are aggregation functions used for?

Consider the following prototypical situations. We have several criteria with respect to which we assess different options (or objects), and every option fulfills each criterion only partially (and has a score on $[0, 1]$ scale). Our aim is to evaluate the combined score for each option (possibly to rank the options).

We may decide to average all the scores. This is a typical approach in sports competitions (like artistic skating) where the criteria are scores given by judges. The total score is the arithmetic mean.

$$f(x_1, \ldots, x_n) = \frac{1}{n}(x_1 + x_2 + \ldots + x_n).$$

We may decide to take a different approach: low scores pull the overall score down. The total score will be no greater than the minimum individual criterion score. This is an example of conjunctive behavior. Conjunctive aggregation functions are suitable to model conjunctions like

If x *is* A **AND** y *is* B **AND** z *is* C **THEN ...**

where A, B and C are the criteria against which the parameters x, y, z are assessed. For example, this is how they choose astronauts: the candidates must fulfill this and that criteria, and having an imperfect score in just one criterion moves the total to that imperfect score, and further down if there is more than one imperfect score.

Yet another approach is to let high scores push each other up. We start with the largest individual score, and then each other nonzero score pushes the total up. The overall score will be no smaller than the maximum of all individual scores. This is an example of disjunctive behavior. Such aggregation functions model disjunction in logical rules like

If x *is* A **OR** y *is* B **OR** z *is* C **THEN ...**

For example, consider collecting evidence supporting some hypothesis. Having more than one piece of supporting evidence makes the total support stronger than the support due to any single piece of evidence. A simple example may be: if you have fever you may have a cold. If you cough and sneeze, you may have a cold. But if you have fever, cough and sneeze at the same time, you are almost certain to have a cold.

It is also possible to think of an aggregation scheme where low scores pull each other down and high scores push each other up. We need to set some threshold to distinguish low and high scores, say 0.5. Then an aggregation function will have conjunctive behavior when all $x_i \leq 0.5$, disjunctive behavior when all $x_i \geq 0.5$, and either conjunctive, disjunctive or averaging behavior when there are some low and some high scores.

One typical use of such aggregation functions is when low scores are interpreted as "negative" information, and high scores as "positive" information. Sometimes scientists also use a bipolar scale ($[-1, 1]$ instead of $[0, 1]$, in which case the threshold is 0).

Multiple attribute decision making

In problems of multiple attribute decision making (sometimes interchangeably called multiple criteria decision making [5]) [55, 132, 133], an alternative (a decision) has to be chosen based on several, usually conflicting criteria. The alternatives are evaluated by using attributes, or features, which are expressed numerically[6]. For example, when buying a car, the attributes are usually the price, quality, fuel consumption, size, power, brand (a nominal attribute), etc. In order to choose the best alternative, one needs to combine the values of the attributes in some way. One popular approach is called *Multi-Attribute Utility Theory* (MAUT). It assigns a numerical score to each alternative, called its *utility* $u(a_1, \ldots, a_n)$. Values a_i denote the numerical scores of each attribute i. Note that a bigger value of a_i does not imply "better", for example when choosing a car, one may prefer medium-sized cars.

The basic assumption of MAUT is that the total utility is a function of individual utilities $x_i = u_i(a_i)$, i.e., $u(\mathbf{a}) = u(u_1(a_1), \ldots, u_n(a_n)) = u(\mathbf{x})$. The

[5] Multiple criteria decision making, besides multiple attribute decision making, also involves multiple objective decision making [132].

[6] Of course, sometimes attributes are qualitative, or ordinal. They may be converted to a numerical scale, e.g., using utility functions.

individual utilities depend *only* on the corresponding attribute a_i, and not on the other attributes, although the attributes can be correlated of course. This is a simplifying assumption, but it allows one to model quite accurately many practical problems.

The rational decision-making axiom implies that one cannot prefer an alternative which differs from another alternative in that it is inferior with respect to some individual utilities, but not superior with respect to the other ones. Mathematically, this means that the function u is monotone non-decreasing with respect to all arguments. If we scale the utilities to $[0, 1]$, and add the boundary conditions $u(0, \ldots, 0) = 0$, $u(1, \ldots, 1) = 1$, we obtain that u is an aggregation function.

The most common aggregation functions used in MAUT are the additive and multiplicative utilities, which are the weighted arithmetic and geometric means (see p. 24). However disjunctive and conjunctive methods are also popular [55]. Compensation is an important notion in MAUT, expressing the concept of trade-off. It implies that the decrease of the total utility u due to some attributes can be compensated by the increase due to other attributes. For example, when buying an item, the increase in price can be compensated by the increase in quality.

Group decision making

Consider a group of n experts who evaluate one (or more) alternatives. Each expert expresses his/her evaluation on a numerical scale (which is the strength of this expert's preference). The goal is to combine all experts' evaluations into a single score [86, 87, 131]. By scaling the preferences to $[0, 1]$ we obtain a vector of inputs $\mathbf{x} = (x_1, \ldots, x_n)$, where x_i is the degree of preference of the i-th expert. The overall evaluation will be some value $y = f(\mathbf{x})$.

The most commonly used aggregation method in group decision making is the weighted arithmetic mean. If experts have different standing, then their scores are assigned different weights. However, experts' opinions may be correlated, and one has to model their interactions and groupings. Further, one has to model such concepts as dictatorship, veto, oligarchies, etc.

From the mathematical point of view, the function f needs to be monotone non-decreasing (the increase in one expert's score cannot lead to the decrease of the overall score) and satisfy the boundary conditions. Thus f is an aggregation function.

Fuzzy logic and rule based systems

In fuzzy set theory [280], membership of objects in fuzzy sets are numbers from $[0, 1]$. Fuzzy sets allow one to model vagueness and uncertainty which are very often present in natural languages. For example the set "ripe bananas" is fuzzy, as there are obviously different degrees of ripeness. Similarly, the sets of "small numbers", "tall people", "high blood pressure" are fuzzy, since there is

no clear cutoff which discriminates objects between those that are in the set and those that are not. An object may simultaneously belong to a fuzzy set and its complement. A fuzzy set A defined on a set of objects X is represented by a membership function $\mu_A : X \to [0, 1]$, in such a way that for any object $x \in X$ the value $\mu_A(x)$ measures the degree of membership of x in the fuzzy set A.

The classical operations of fuzzy sets, intersection and union, are based on the minimum and maximum, i.e., $\mu_{A\cap B} = \min\{\mu_A, \mu_B\}$, $\mu_{A\cup B} = \max \{\mu_A, \mu_B\}$. However other aggregation functions, such as the product, have been considered almost since the inception of fuzzy set theory [34]. Nowadays a large class of conjunctive and disjunctive functions, the triangular norms and conorms, are used to model fuzzy set intersection and union.

Fuzzy set theory has proved to be extremely useful for solving many real world problems, in which the data are imprecise, e.g., [81, 86, 275, 284, 285, 286]. Fuzzy control in industrial systems and consumer electronics is one notable example of the practical applications of fuzzy logic.

Rule based systems, especially fuzzy rule based systems [285], involve aggregation of various numerical scores, which correspond to degrees of satisfaction of rule antecedents. A rule can be a statement like

If x *is* A **AND** y *is* B **AND** z *is* C **THEN** some action

The antecedents are usually membership values of x in A, y in B, etc. The strength of "firing" the rule is determined by an aggregation function that combines membership values $f(\mu_A(x), \mu_B(y), \mu_C(z))$.

There are many other uses of aggregation functions, detailed in the references at the end of this Chapter.

1.3 Classification and general properties

1.3.1 Main classes

As we have seen in the previous section, there are various semantics of aggregation, and the main classes are determined according to these semantics. In some cases we require that high and low inputs average each other, in other cases aggregation functions model logical connectives (disjunction and conjunction), so that the inputs reinforce each other, and sometimes the behavior of aggregation functions depends on the inputs. The four main classes of aggregation functions are [43, 81, 83, 86, 87]

- Averaging,
- Conjunctive,
- Disjunctive,
- Mixed.

Definition 1.7 (Averaging aggregation). *An aggregation function f has averaging behavior (or is averaging) if for every $\mathbf{x}$ it is bounded by*

$$\min(\mathbf{x}) \leq f(\mathbf{x}) \leq \max(\mathbf{x}).$$

Definition 1.8 (Conjunctive aggregation). *An aggregation function f has conjunctive behavior (or is conjunctive) if for every $\mathbf{x}$ it is bounded by*

$$f(\mathbf{x}) \leq \min(\mathbf{x}) = \min(x_1, x_2, \ldots, x_n).$$

Definition 1.9 (Disjunctive aggregation). *An aggregation function f has disjunctive behavior (or is disjunctive) if for every $\mathbf{x}$ it is bounded by*

$$f(\mathbf{x}) \geq \max(\mathbf{x}) = \max(x_1, x_2, \ldots, x_n).$$

Definition 1.10 (Mixed aggregation). *An aggregation function f is mixed if it does not belong to any of the above classes, i.e., it exhibits different types of behavior on different parts of the domain.*

1.3.2 Main properties

Definition 1.11 (Idempotency). *An aggregation function f is called idempotent if for every input $\mathbf{x} = (t, t, \ldots, t), t \in [0, 1]$ the output is $f(t, t \ldots, t) = t$.*

Note 1.12. Because of monotonicity of f, idempotency is equivalent to averaging behavior.[7]

The aggregation functions *minimum* and *maximum* are the only two functions that are at the same time conjunctive (disjunctive) and averaging, and hence idempotent.

Example 1.13. The arithmetic mean is an averaging (idempotent) aggregation function

$$f(\mathbf{x}) = \frac{1}{n}(x_1 + x_2 + \ldots + x_n).$$

Example 1.14. The geometric mean is also an averaging (idempotent) aggregation function

$$f(\mathbf{x}) = \sqrt[n]{x_1 x_2 \ldots x_n}.$$

[7] Proof: Take any $\mathbf{x} \in [0, 1]^n$, and denote by $p = \min(\mathbf{x}), q = \max(\mathbf{x})$. By monotonicity, $p = f(p, p, \ldots, p) \leq f(\mathbf{x}) \leq f(q, q, \ldots, q) = q$. Hence $\min(\mathbf{x}) \leq f(\mathbf{x}) \leq \max(\mathbf{x})$. The converse: let $\min(\mathbf{x}) \leq f(\mathbf{x}) \leq \max(\mathbf{x})$. By taking $\mathbf{x} = (t, t, \ldots, t)$, $\min(\mathbf{x}) = \max(\mathbf{x}) = f(\mathbf{x}) = t$, hence idempotency.

Example 1.15. The product is a conjunctive aggregation function

$$f(\mathbf{x}) = \prod_{i=1}^{n} x_i = x_1 x_2 \ldots x_n.$$

Definition 1.16 (Symmetry). *An aggregation function f is called symmetric, if its value does not depend on the permutation of the arguments, i.e.,*

$$f(x_1, x_2, \ldots, x_n) = f(x_{P(1)}, x_{P(2)}, \ldots, x_{P(n)}),$$

for every $\mathbf{x}$ and every permutation $P = (P(1), P(2), \ldots, P(n))$ of $(1, 2 \ldots, n)$.

The semantical interpretation of symmetry is anonymity, or equality. For example, equality of judges in sports competitions: all inputs are treated equally, and the output does not change if the judges swap seats.[8] On the other hand, in shareholders meetings the votes are not symmetric as they depend on the number of shares each voter has.

Example 1.17. The arithmetic and geometric means and the product in Examples 1.13-1.15 are symmetric aggregation functions. A weighted arithmetic mean with non-equal weights $w_1, w_2, \ldots w_n$, that are non-negative and add to one is not symmetric,

$$f(\mathbf{x}) = \sum_{i=1}^{n} w_i x_i = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n.$$

Permutation of arguments is very important in aggregation, as it helps express symmetry, as well as to define other concepts. A permutation of $(1, 2 \ldots, 5)$ is just a tuple like $(5, 3, 2, 1, 4)$. There are $n! = 1 \times 2 \times 3 \times \ldots \times n$ possible permutations of $(1, 2, \ldots, n)$.

We will denote a vector whose components are arranged in the order given by a permutation P by $\mathbf{x}_P = (x_{P(1)}, x_{P(2)}, \ldots, x_{P(n)})$. In our example, $\mathbf{x}_P = (x_5, x_3, x_2, x_1, x_4)$. We will frequently use the following special permutations of the components of $\mathbf{x}$.

Definition 1.18. *We denote by $\mathbf{x}_{\nearrow}$ the vector obtained from $\mathbf{x}$ by arranging its components in non-decreasing order, that is, $\mathbf{x}_{\nearrow} = \mathbf{x}_P$ where P is the permutation such that $x_{P(1)} \leq x_{P(2)} \leq \ldots \leq x_{P(n)}$.*

Similarly, we denote by $\mathbf{x}_{\searrow}$ the vector obtained from $\mathbf{x}$ by arranging its components in non-increasing order, that is $\mathbf{x}_{\searrow} = \mathbf{x}_P$ where P is the permutation such that $x_{P(1)} \geq x_{P(2)} \geq \ldots \geq x_{P(n)}$.

---

[8] It is frequently interpreted as anonymity criterion: anonymous ballot papers can be counted in any order.

Note 1.19. In fuzzy sets literature, the notation $\mathbf{x}_{()} = (x_{(1)}, \ldots, x_{(n)})$ is often used to denote both $\mathbf{x}_{\nearrow}$ and $\mathbf{x}_{\searrow}$, depending on the context.

Note 1.20. We can express the symmetry property by an equivalent statement that for every input vector $\mathbf{x}$

$$f(\mathbf{x}) = f(\mathbf{x}_{\nearrow}) \ \ (\text{ or } f(\mathbf{x}) = f(\mathbf{x}_{\searrow})),$$

rather than $f(\mathbf{x}) = f(\mathbf{x}_P)$ for *every* permutation. This gives us a shortcut for calculating the value of a symmetric aggregation function for a given $\mathbf{x}$ by using `sort()` operation (see Fig. 1.1).

```
#include<algorithm>
struct Greaterthan {
   bool operator()(const double& a, const double& b) {return a > b; }
} greaterthan; /* required to specify the user's sorting order */

double f_symm(int n, double * x)
{
  sort( &x[0], &x[n], greaterthan); /* sorted in decreasing order */
  /* evaluate f for x sorted in decreasing order */
  return f(n,x);
}
/* to sort in increasing order, define the structure Lessthan
   in an analogous way. */
```

Fig. 1.1. A C++ code for evaluation of a symmetric aggregation function. $f(\mathbf{x})$ is defined for $\mathbf{x}_{\searrow}$, $f\_symm(\mathbf{x})$ returns a correct value for any $\mathbf{x}$.

Definition 1.21 (Strict monotonicity). *An aggregation function f is strictly monotone increasing if*

$$\mathbf{x} \leq \mathbf{y} \text{ but } \mathbf{x} \neq \mathbf{y} \text{ implies } f(\mathbf{x}) < f(\mathbf{y}) \text{ for every } \mathbf{x}, \mathbf{y} \in [0,1]^n. \tag{1.3}$$

Note 1.22. Notice the difference between $[\mathbf{x} \leq \mathbf{y}, \mathbf{x} \neq \mathbf{y}]$ and $\mathbf{x} < \mathbf{y}$. The latter implies that for *all* components of $\mathbf{x}$ and $\mathbf{y}$ we have $x_i < y_i$, whereas the former means that at least one component of $\mathbf{y}$ is greater than that of $\mathbf{x}$, i.e., $\exists i$ such that $x_i < y_i$ and $\forall j : x_j \leq y_j$.

Strict monotonicity is a rather restrictive property. Note that there are no strictly monotone conjunctive or disjunctive aggregation functions. This is because every conjunctive function coincides with $\min(\mathbf{x})$ for those $\mathbf{x}$ that have at least one zero component, and min is not strictly monotone (similarly, disjunctive aggregation functions coincide with $\max(\mathbf{x})$ for those $\mathbf{x}$ that have at least one component $x_i = 1$). However, strict monotonicity on the semi-open set $]0,1]^n$ (respectively $[0,1[^n$) is often considered for conjunctive (disjunctive)

aggregation functions, see Chapter 3. Of course, there are plenty of strictly increasing averaging aggregation functions, such as arithmetic means.

It is often the case that when an input vector contains a specific value, this value can be omitted. For example, consider a conjunctive aggregation function f to model the rule for buying bananas

If *price is LOW* **AND** *banana is RIPE* **THEN** buy it.

Conjunction means that we only want cheap and ripe bananas, but both are matters of degree, expressed on $[0,1]$ scale. If one of the arguments $x_i = 1$, e.g., we find perfectly ripe bananas, then the outcome of the rule is equal to the degree of our satisfaction with the price.

Definition 1.23 (Neutral element). *An aggregation function f has a neutral element $e \in [0,1]$, if for every $t \in [0,1]$ in any position it holds*

$$f(e,\ldots,e,t,e,\ldots,e) = t.$$

For extended aggregation functions, we have a stronger version of this property, which relates aggregation functions with a different number of arguments.

Definition 1.24 (Strong neutral element). *An extended aggregation function F has a neutral element $e \in [0,1]$, if for every $\mathbf{x}$ with $x_i = e$, for some $1 \le i \le n$, and every $n \ge 2$,*

$$f_n(x_1,\ldots,x_{i-1},e,x_{i+1},\ldots,x_n) = f_{n-1}(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n),$$

When $n = 2$, we have $f(t,e) = f(e,t) = t$. Then by iterating this property we obtain as a consequence that every member f_n of the family has the neutral element e, i.e.,

$$f_n(e,\ldots,e,t,e,\ldots,e) = t,$$

for t in any position.

Note 1.25. A neutral element, if it exists, is unique [9]. It can be any number from $[0,1]$.

Note 1.26. Observe that if an aggregation function f has neutral element $e = 1$ (respectively $e = 0$) then f is necessarily conjunctive (respectively disjunctive). Indeed, if f has neutral element $e = 1$, then by monotonicity it is $f(x_1,\ldots,x_n) \le f(1,\ldots,1,x_i,1,\ldots,1) = x_i$ for any $i \in \{1,\ldots,n\}$, and this implies $f \le \min$ (the proof for the case $e = 0$ is analogous).

[9] Proof: Assume f has two neutral elements e and u. Then $u = f(e,u) = e$, therefore $e = u$. For n variables, assume $e < u$. By monotonicity, $e = f(e,u,\ldots,u,\ldots,u) \ge f(e,e,\ldots,e,u,e\ldots,e) = u$, hence we have a contradiction. The case $e > u$ leads to a similar contradiction.

Note 1.27. The concept of a neutral element has been recently extended to that of neutral tuples, see [29].

Example 1.28. The product function $f(\mathbf{x}) = \prod x_i$ has neutral element $e = 1$. Similarly, min function has neutral element $e = 1$ and max function has neutral element $e = 0$. The arithmetic mean does not have a neutral element. We shall see later on that any triangular norm has $e = 1$, and any triangular conorm has $e = 0$.

It may also be the case that one specific value a of any argument yields the output a. For example, if we use conjunction for aggregation, then if any input is 0, then the output must be 0 as well. In the banana buying rule above, if the banana is green, we do not buy it at any price.

Definition 1.29 (Absorbing element (annihilator)). *An aggregation function f has an absorbing element $a \in [0,1]$ if*

$$f(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_n) = a,$$

for every $\mathbf{x}$ such that $x_i = a$ with a in any position.

Note 1.30. An absorbing element, if it exists, is unique. It can be any number from $[0,1]$.

Example 1.31. Any conjunctive aggregation function has absorbing element $a = 0$. Any disjunctive aggregation function has absorbing element $a = 1$. This is a simple consequence of the Definitions 1.8 and 1.9. Some averaging functions also have an absorbing element, for example the geometric mean

$$f(\mathbf{x}) = \left(\prod_{i=1}^{n} x_i \right)^{1/n}$$

has the absorbing element $a = 0$.

Note 1.32. An aggregation function with an annihilator in $]0,1[$ cannot have a neutral element[10]. But it may have a neutral element if $a = 0$ or $a = 1$.

Note 1.33. The concept of an absorbing element has been recently extended to that of absorbing tuples, see [28].

---

[10] Proof: Suppose $a \in]0,1[$ is the absorbing element and $e \in [0,1]$ is the neutral element. Then if $a \leq e$, we get the contradiction $a = 0$, since it is $a = f(a, \ldots, a, 0) \leq f(e, \ldots, e, 0) = 0$. Similarly, if $a > e$ then $a = f(a, \ldots, a, 1) \geq f(e, \ldots, e, 1) = 1$.

Definition 1.34 (Zero divisor). *An element $a \in]0,1[$ is a zero divisor of an aggregation function f if for all $i \in \{1,\dots,n\}$ there exists some $\mathbf{x} \in]0,1]^n$ such that its i-th component is $x_i = a$, and it holds $f(\mathbf{x}) = 0$, i.e., the equality*

$$f(x_1,\dots,x_{i-1},a,x_{i+1},\dots,x_n) = 0,$$

can hold for some $\mathbf{x} > \mathbf{0}$ with a at any position.

Note 1.35. Because of monotonicity of f, if a is a zero divisor, then all values $b \in]0,a]$ are also zero divisors.

The interpretation of zero divisors is straightforward: if one of the inputs takes the value a, or a smaller value, then the aggregated value could be zero, for some $\mathbf{x}$. So it is possible to have the aggregated value zero, even if all the inputs are positive. The largest value a (or rather an upper bound on a) plays the role of a threshold, the lower bound on all the inputs which guarantees a non-zero output. That is, if b is *not* a zero divisor, then $f(\mathbf{x}) > 0$, if all $x_i \geq b$.

Example 1.36. Averaging aggregation functions do not have zero divisors. But the function $f(x_1,x_2) = \max\{0, x_1 + x_2 - 1\}$ has a zero divisor $a = 0.999$, which means that the output can be zero even if any of the components x_1 or x_2 is as large as 0.999, provided that the other component is sufficiently small. However, 1 is not a zero divisor.

Zero divisors exist for aggregation functions that exhibit conjunctive behavior, at least on parts of their domain, i.e., conjunctive and mixed aggregation functions. For disjunctive aggregation functions we have an analogous definition.

Definition 1.37 (One divisor). *An element $a \in]0,1[$ is a one divisor of an aggregation function f if for all $i = 1,\dots,n$ there exists some $\mathbf{x} \in [0,1[^n$ such that its i-th component is $x_i = a$ and it holds $f(\mathbf{x}) = 1$, i.e., the equality*

$$f(x_1,\dots,x_{i-1},a,x_{i+1},\dots,x_n) = 1,$$

can hold for some $\mathbf{x} < \mathbf{1}$ with a at any position.

The interpretation is similar: the value of any inputs larger than a can make the output $f(\mathbf{x}) = 1$, even if none of the inputs is actually 1. On the other hand, if b is not a one divisor, then the output cannot be one if all the inputs are no larger than b.

The following property is useful for construction of n-ary aggregation functions from a single two-variable function.

Definition 1.38 (Associativity). *A two-argument function f is associative if $f(f(x_1,x_2),x_3) = f(x_1, f(x_2,x_3))$ holds for all x_1,x_2,x_3 in its domain.*

Consequently, the n-ary aggregation function can be constructed in a unique way by iteratively applying f_2 as

$$f_n(x_1,\ldots,x_n) = f_2(f_2(\ldots f_2(x_1,x_2),x_3),\ldots,x_n).$$

Thus bivariate associative aggregation functions univocally define extended aggregation functions.

Example 1.39. The product, minimum and maximum are associative aggregation functions. The arithmetic mean is not associative.

Associativity simplifies calculation of aggregation functions, and it effectively allows one to easily aggregate any number of inputs, as the following code on Fig. 1.2 illustrates. It is not the only way of doing this (for instance the arithmetic or geometric means are also easily computed for any number of inputs).

Another construction gives what are called recursive extended aggregation functions by Montero [68]. It involves a family of two-variate functions $f_2^n, n = 2, 3, \ldots$.

Definition 1.40 (Recursive extended aggregation function). *An extended aggregation function F is recursive by Montero if the members f_n are defined from a family of two-variate aggregation functions f_2^n recursively as*

$$f_n(x_1,\ldots,x_n) = f_2^n(f_{n-1}(x_1,\ldots,x_{n-1}),x_n),$$

starting with $f_2 = f_2^2$.

Each extended aggregation function built from an associative bivariate aggregation function is recursive by Montero, but the converse is not true.

Example 1.41. Define $f_2^n(t_1,t_2) = \frac{(n-1)t_1+t_2}{n}$. Then $f_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i$, the arithmetic mean (which is not associative).

Definition 1.42 (Decomposable extended aggregation function). *An extended aggregation function F is decomposable if for all $m,n = 1,2,\ldots$ and for all $\mathbf{x} \in [0,1]^m$, $\mathbf{y} \in [0,1]^n$:*

$$f_{m+n}(x_1,\ldots,x_m,y_1,\ldots,y_n) = \\ f_{m+n}(\underbrace{f_m(x_1,\ldots,x_m),\ldots,f_m(x_1,\ldots,x_m)}_{m \text{ times}},y_1,\ldots,y_n). \tag{1.4}$$

```
/* Input: the vector of arguments x, two-variable function f
   Output: the value f_n(x).*/
// Recursive method
    double fn(int n, double[] x, (double)f(double x, double y))
{
   if(n==2) return f(x[0],x[1]);
   else return f(fn(n-1,x,f),x[n-1]);
}

// Non-recursive method
double fn(int n, double[] x, (double)f(double x, double y))
{
   double s=f(x[0],x[1]);
   for(i=2;i<n;i++) s=f(s,x[i]);
   return s;
}
```

Fig. 1.2. Recursive and non-recursive calculation of an associative function.

A continuous decomposable extended aggregation function is always idempotent.

Another useful property, which generalizes both symmetry and associativity, and is applicable to extended aggregation functions, is called bisymmetry. Consider the situation in which m jurymen evaluate an alternative with respect to n criteria. Let $x_{ij}, i = 1, \dots, m, j = 1, \dots, n$ denote the score given by the i-th juryman with respect to the j-th criterion. To compute the global score $f_{mn}(x_{11}, \dots, x_{1n}, \dots, x_{mn})$ we can either evaluate the scores given by the i-th juryman, $y_i = f_n(x_{i1}, \dots, x_{in})$, and then aggregate them as $z = f_m(y_1, \dots, y_m)$, or, alternatively, aggregate scores of all jurymen with respect to each individual criterion j, i.e., compute $\tilde{y}_j = f_m(x_{1j}, \dots, x_{mj})$, and then aggregate these scores as $\tilde{z} = f_n(\tilde{y}_1, \dots, \tilde{y}_n)$. The third alternative is to aggregate all the scores by an aggregation function $f_{mn}(\mathbf{x})$.

This is illustrated in Table 1.1. We can either aggregate scores in each row, and then aggregate the totals in the last column of this table, or we can aggregate scores in each column, and then aggregate the totals in the last row, or aggregate all scores at once. The bisymmetry property simply means that all three methods lead to the same answer.

Definition 1.43 (Bisymmetry). *An extended aggregation function F is bisymmetric if for all $m, n = 1, 2, \dots$ and for all $\mathbf{x} \in [0,1]^{mn}$:*

$$\begin{aligned} f_{mn}(\mathbf{x}) &= f_m(f_n(x_{11}, \dots, x_{1n}), \dots, f_n(x_{m1}, \dots, x_{mn})) \\ &= f_n(f_m(x_{11}, \dots, x_{m1}), \dots, f_m(x_{1n}, \dots, x_{mn})). \end{aligned} \tag{1.5}$$

Table 1.1. The table of scores to be aggregated by m jurymen with respect to n criteria.

juror \ criterion	1	2	3	...	n	Total
1	x_{11}	x_{12}	x_{13}	...	x_{1n}	y_1
2	x_{21}	x_{22}	x_{23}	...	x_{2n}	y_2
3	x_{31}	x_{32}	x_{33}	...	x_{3n}	y_3
4	x_{41}	x_{42}	x_{43}	...	x_{4n}	y_4
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
m	x_{m1}	x_{m2}	x_{m3}	...	x_{mn}	y_m
Total	$\tilde{y}_1$	$\tilde{y}_2$	$\tilde{y}_3$	...	$\tilde{y}_n$	$\tilde{z}\backslash z$

Note 1.44. A symmetric associative extended aggregation function is bisymmetric. However there are symmetric and bisymmetric non-associative extended aggregation functions, e.g., the arithmetic and geometric means. The extended aggregation function defined by $f(\mathbf{x}) = x_1$ (projection to the first coordinate) is bisymmetric and associative but not symmetric. The extended aggregation function $f(\mathbf{x}) = \left(\sum_{i=1}^{n} \frac{x_i}{n}\right)^2$ (square of the arithmetic mean) is symmetric but neither associative nor bisymmetric. Every continuous associative extended aggregation function is bisymmetric, but not necessarily symmetric.

Let us finally mention two properties describing the stability of aggregation functions with respect to some changes of the scale:

Definition 1.45 (Shift-invariance). *An aggregation function $f : [0,1]^n \to [0,1]$ is shift-invariant (or stable for translations) if for all $\lambda \in [-1,1]$ and for all $(x_1,\ldots,x_n) \in [0,1]^n$ it is*

$$f(x_1+\lambda,\ldots,x_n+\lambda) = f(x_1,\ldots,x_n)+\lambda$$

whenever $(x_1+\lambda,\ldots,x_n+\lambda) \in [0,1]^n$ and $f(x_1,\ldots,x_n)+\lambda \in [0,1]$.

Definition 1.46 (Homogeneity). *An aggregation function $f : [0,1]^n \to [0,1]$ is homogeneous if for all $\lambda \in [0,1]$ and for all $(x_1,\ldots,x_n) \in [0,1]^n$ it is*

$$f(\lambda x_1,\ldots,\lambda x_n) = \lambda f(x_1,\ldots,x_n).$$

Aggregation functions which are both shift-invariant and homogeneous are known as *linear aggregation functions.*

Note that, due to the boundary conditions $f(0,\ldots,0)=0$ and $f(1,\ldots,1)=1$, either shift-invariant, homogeneous or linear aggregation functions are necessarily idempotent, and thus (see Note 1.12) they can only be found among averaging functions. A prototypical example of a linear aggregation function is the arithmetic mean.

1.3.3 Duality

It is often useful to draw a parallel between conjunctive and disjunctive aggregation functions, as they often satisfy very similar properties, just viewed from a different angle. The concept of a dual aggregation function helps with mapping most properties of conjunctive aggregation functions to disjunctive ones. So essentially one studies conjunctive functions, and obtains the corresponding results for disjunctive functions by duality. There are also aggregation functions that are self-dual.

First we need the concept of negation.

Definition 1.47 (Strict negation). *A univariate function N defined on $[0,1]$ is called a strict negation, if its range is also $[0,1]$ and it is strictly monotone decreasing.* [11]

Definition 1.48 (Strong negation). *A univariate function N defined on $[0,1]$ is called a strong negation, if it is strictly decreasing and involutive (i.e., $N(N(t)) = t$ for all $t \in [0,1]$).*

Example 1.49. The most commonly used strong negation is the standard negation $N(t) = 1 - t$. We will use it throughout this book. Another example of negation is $N(t) = 1 - t^2$, which is strict but not strong.

Note 1.50. A strictly monotone bijection is always continuous. Hence strict and strong negations are continuous.

Despite its simplicity, the standard negation plays a fundamental role in the construction of strong negations, since any strong negation can be built from the standard negation using an automorphism[12] of the unit interval [239]:

Theorem 1.51. *A function $N : [0,1] \to [0,1]$ is a strong negation if and only if there exists an automorphism $\varphi : [0,1] \to [0,1]$ such that $N = N_\varphi = \varphi^{-1} \circ (1 - Id) \circ \varphi$, i.e. $N(t) = N_\varphi(t) = \varphi^{-1}(1 - \varphi(t))$ for any $t \in [0,1]$.*

Example 1.52. Let us construct some strong negations:

- With $\varphi(a) = a^\lambda$ $(\lambda > 0)$:

$$N_\varphi(t) = (1 - t^\lambda)^{1/\lambda}$$

  (note that the standard negation is recovered with $\lambda = 1$);

---

[11] A frequently used term is *bijection*: a bijection is a function $f : A \to B$, such that for every $y \in B$ there is exactly one $x \in A$, such that $y = f(x)$, i.e., it defines a one-to-one correspondence between A and B. Because N is strictly monotone, it is a one-to-one function. Its range is [0,1], hence it is an *onto* mapping, and therefore a bijection.

[12] *Automorphism* is another useful term: An automorphism is a strictly increasing bijection of an interval onto itself $[a,b] \to [a,b]$.

- With $\varphi(a) = 1 - (1-a)^\lambda$ ($\lambda > 0$):
$$N_\varphi(t) = 1 - [1 - (1-t)^\lambda]^{1/\lambda}$$
(the standard negation is recovered with $\lambda = 1$);
- With $\varphi(a) = \frac{a}{\lambda + (1-\lambda)a}$ ($\lambda > 0$):
$$N_\varphi(t) = \frac{\lambda^2(1-t)}{t + \lambda^2(1-t)}$$
(again, the standard negation is obtained with $\lambda = 1$);

- With $\varphi(a) = \frac{\ln(1+\lambda a^\alpha)}{\ln(1+\lambda)}$, ($\lambda > -1, \alpha > 0$):
$$N_\varphi(t) = \left(\frac{1-t^\alpha}{1+\lambda t^\alpha}\right)^{1/\alpha}.$$
Note that taking $\alpha = 1$ we get the family $N_\varphi(t) = \frac{1-t}{1+\lambda t}$, which is known as the Sugeno's family of strong negations (which includes, when $\lambda = 0$, the standard negation).

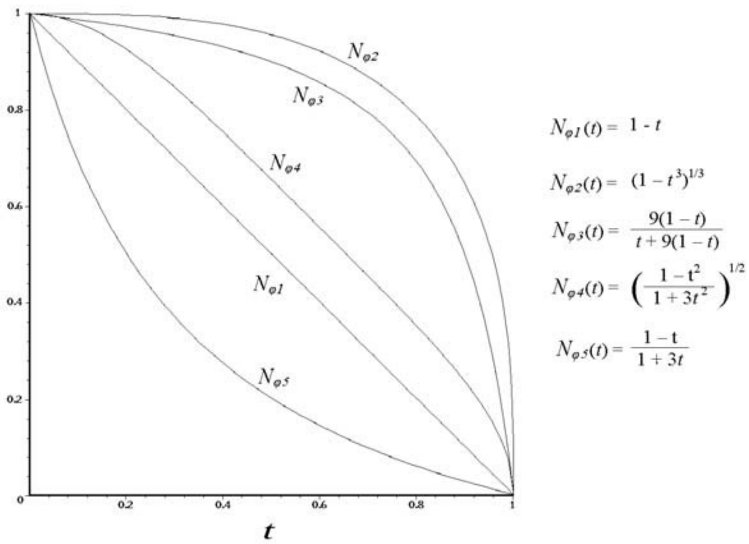


Fig. 1.3. Graphs of some strong negations in Example 1.52 with a fixed parameter λ.

Note 1.53. The characterization given in Theorem 1.51 allows one to easily show that any strong negation N has a unique *fixed point*, i.e., there exists one and only one value in $[0,1]$, which we will denote t_N, verifying $N(t_N) = t_N$. Indeed, since $N = N_\varphi$ for some automorphism φ, the equation $N(t_N) = t_N$ is equivalent to $\varphi^{-1}(1-\varphi(t_N)) = t_N$, whose unique solution is given by $t_N = \varphi^{-1}(1/2)$. Note that, obviously, it is always $t_N \neq 0$ and $t_N \neq 1$.

Definition 1.54 (Dual aggregation function). *Let $N : [0,1] \to [0,1]$ be a strong negation and $f : [0,1]^n \to [0,1]$ an aggregation function. Then the aggregation function f_d given by*

$$f_d(x_1,\ldots,x_n) = N(f(N(x_1), N(x_2), \ldots, N(x_n)))$$

is called the dual of f with respect to N, or, for short, the N-dual of f. When using the standard negation, f_d is given by

$$f_d(x_1,\ldots,x_n) = 1 - f(1-x_1,\ldots,1-x_n)$$

and we will simply say that f_d is the dual of f.

It is evident that the dual of a conjunctive aggregation function is disjunctive, and vice versa, regardless of what strong negation is used. Some functions are *self-dual.*

Definition 1.55 (Self-dual aggregation function). *Given a strong negation N, an aggregation function f is self-dual with respect to N (for short, N-self-dual or N-invariant), if*

$$f(\mathbf{x}) = N(f(N(\mathbf{x}))),$$

where $N(\mathbf{x}) = (N(x_1),\ldots,N(x_n))$. For the standard negation we have

$$f(\mathbf{x}) = 1 - f(\mathbf{1} - \mathbf{x}),$$

and it is simply said that f is self-dual.

For example, the arithmetic mean is self-dual. We study N-self-dual aggregation functions in detail in Chapter 4. It is worth noting that there are no N-self-dual conjunctive or disjunctive aggregation functions.

1.3.4 Comparability

Sometimes it is possible to compare different aggregation functions and establish a certain order among them. We shall compare aggregation functions pointwise, i.e., for every $\mathbf{x} \in [0,1]^n$.

Definition 1.56. *An aggregation function f is stronger than another aggregation function of the same number of arguments g, if for all $\mathbf{x} \in [0,1]^n$:*

$$g(\mathbf{x}) \leq f(\mathbf{x}).$$

It is expressed as $g \leq f$. When f is stronger that g, it is equivalently said that g is weaker than f.

Not all aggregation functions are comparable. It may happen that f is stronger than g only on some part of the domain, and the opposite is true on the rest of the domain. In this case we say that f and g are *incomparable*.

Example 1.57. The strongest conjunctive aggregation function is the *minimum*, and the weakest disjunctive aggregation function is the *maximum* (see Definitions 1.8 and 1.9). Any disjunctive aggregation function is stronger than an averaging function, and any averaging function is stronger than a conjunctive one.

1.3.5 Continuity and stability

We will be mostly interested in *continuous* aggregation functions, which intuitively are such functions that a small change in the input results in a small change in the output. [13] There are some interesting aggregation functions that are discontinuous, but from the practical point of view continuity is very important for producing a stable output.

The next definition is an even stronger continuity requirement. The reason is that simple, or even uniform continuity is not sufficient to distinguish functions that produce a "small" change in value due to a small change of the argument [14]. The following definition puts a bound on the actual change in value due to changes in the input.

[13] A real function of n arguments is continuous if for any sequences $\{x_{ij}\}, i = 1,\ldots,n$ such that $\lim_{j\to\infty} x_{ij} = y_i$ it holds $\lim_{j\to\infty} f(x_{1j},\ldots x_{nj}) = f(y_1,\ldots,y_n)$. Because the domain $[0,1]^n$ is a compact set, continuity is equivalent to its stronger version, uniform continuity. For monotone functions we have a stronger result: an aggregation function is uniformly continuous if and only if it is continuous in each argument (i.e., we can check continuity by fixing all variables but one, and checking continuity of each univariate function. However, general non-monotone functions can be continuous in each variable without being continuous).

[14] Think of this: a discontinuous (but integrable) function can be approximated arbitrarily well by some continuous function (e.g., a polynomial). Thus based on their values, or graphs, we cannot distinguish between continuous and discontinuous integrable functions, as the values of both functions coincide up to a tiny difference (which we can make as small as we want). A computer will not see any difference between the two types of functions. Mathematically speaking, the subset of continuous functions $C(\Omega)$ is *dense* in the set of integrable functions $L^1(\Omega)$ on a compact set.

Definition 1.58 (Lipschitz continuity). *An aggregation function f is called Lipschitz continuous if there is a positive number M, such that for any two vectors $\mathbf{x}, \mathbf{y}$ in the domain of definition of f:*

$$|f(\mathbf{x}) - f(\mathbf{y})| \leq M d(\mathbf{x}, \mathbf{y}), \tag{1.6}$$

where $d(\mathbf{x}, \mathbf{y})$ is a distance between $\mathbf{x}$ and $\mathbf{y}$ [15]*. The smallest such number M is called the Lipschitz constant of f (in the distance d).*

Typically the distance is the Euclidean distance between vectors,

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_n - y_n)^2},$$

but it can be chosen as any norm $d(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}||$ [16]; typically it is chosen as a p-norm. A p-norm, $p \geq 1$ is a function $||\mathbf{x}||_p = \left(\sum\limits_{i=1}^{n} |x_i|^p \right)^{1/p}$, for finite p, and $||\mathbf{x}||_\infty = \max\limits_{i=1\ldots,n} |x_i|$.

Thus, if the change in the input is $\delta = ||\mathbf{x} - \mathbf{y}||$, then the output will change by at most $M\delta$. Hence M can be interpreted as the upper bound on the rate of change of a function. If a function f is differentiable, then M is simply the upper bound on the norm of its gradient. All differentiable functions are necessarily Lipschitz-continuous, but not vice versa. However, any Lipschitz function is differentiable "almost" everywhere[17].

We pay attention to the rate of change of a function because of the ever present input inaccuracies. If the aggregation function receives an inaccurate input $\tilde{\mathbf{x}} = (x_1 + \delta_1, \ldots, x_n + \delta_n)$, contaminated with some noise $(\delta_1, \ldots, \delta_n)$, we do not expect the output $f(\tilde{\mathbf{x}})$ to be substantially different from $f(\mathbf{x})$. The Lipschitz constant M bounds the factor by which the noise is magnified.

Note 1.59. Since $f(\mathbf{0}) = 0$ and $f(\mathbf{1}) = 1$, the Lipschitz constant of any aggregation function is $M \geq 1/||\mathbf{1}||$. For p-norms we have $||\mathbf{1}|| = \sqrt[p]{n \cdot 1} \leq 1$, that is $M \geq n^{-1/p}$, so in principle M can be smaller than 1.

Definition 1.60 (p-stable aggregation functions). *Given $p \geq 1$, an aggregation function is called p-stable if its Lipschitz constant in the p-norm $||\cdot||_p$ is 1. An extended aggregation function is p-stable if it can be represented as a family of p-stable aggregation functions.*

---

[15] A *distance* between objects from a set S is a function defined on $S \times S$, whose values are non-negative real numbers, with the properties: 1) $d(x, y) = 0$ if and only if $x = y$, 2) $d(x, y) = d(y, x)$, and 3) $d(x, z) \leq d(x, y) + d(y, z)$ (triangular inequality). Such distance is called a *metric*.

[16] A *norm* is a function f on a vector space with the properties: 1) $f(\mathbf{x}) > 0$ for all nonzero $\mathbf{x}$ and $f(\mathbf{0}) = 0$, 2) $f(a\mathbf{x}) = |a| f(\mathbf{x})$, and 3)$f(\mathbf{x} + \mathbf{y}) \leq f(\mathbf{x}) + f(\mathbf{y})$.

[17] I.e., it is differentiable on its entire domain, except for a subset of *measure zero*.

Evidently, p-stable aggregation functions do not enhance input inaccuracies, as $|f(\tilde{\mathbf{x}}) - f(\mathbf{x})| \leq ||\tilde{\mathbf{x}} - \mathbf{x}||_p = ||\boldsymbol{\delta}||_p$.

Definition 1.61 (1-Lipschitz aggregation functions). *An aggregation function f is called 1-Lipschitz if it is p-stable with $p = 1$, i.e., for all $\mathbf{x}, \mathbf{y}$:*

$$|f(\mathbf{x}) - f(\mathbf{y})| \leq |x_1 - y_1| + |x_2 - y_2| + \ldots + |x_n - y_n|.$$

Definition 1.62 (Kernel aggregation functions). *An aggregation function f is called kernel if it is p-stable with $p = \infty$, i.e., for all $\mathbf{x}, \mathbf{y}$:*

$$|f(\mathbf{x}) - f(\mathbf{y})| \leq \max_{i=1,\ldots,n} |x_i - y_i|.$$

For kernel aggregation functions, the error in the output cannot exceed the largest error in the input vector.

Note 1.63. If an aggregation function is p-stable for a given $p > 1$, then it is also q-stable for any $1 \leq q < p$. This is because $||\mathbf{x}||_p \leq ||\mathbf{x}||_q$ for all $\mathbf{x}$.

Example 1.64. The product, minimum and maximum are p-stable extended aggregation functions for any p. The arithmetic mean is also p-stable for any p. The geometric mean is not Lipschitz, although it is continuous.[18]

1.4 Main families and prototypical examples

1.4.1 Min and Max

The minimum and maximum functions are the two main aggregation functions that are used in fuzzy set theory and fuzzy logic. This is partly due to the fact that they are the only two operations consistent with a number of set-theoretical properties, and in particular mutual distributivity [33]. These connectives model fuzzy set intersection and union (or conjunction and disjunction).

They are defined for any number of arguments as

$$\min(\mathbf{x}) = \min_{i=1,\ldots,n} x_i, \tag{1.7}$$

$$\max(\mathbf{x}) = \max_{i=1,\ldots,n} x_i. \tag{1.8}$$

---

[18] Take $f(x_1, x_2) = \sqrt{x_1 x_2}$, which is continuous for $x_1, x_2 \geq 0$, and let $x_2 = 1$. $f(t, 1) = \sqrt{t}$ is continuous but not Lipschitz. To see this, let $t = 0$ and $u > 0$. Then $|\sqrt{0} - \sqrt{u}| = \sqrt{u} > Mu = M|0 - u|$, or $u^{-\frac{1}{2}} > M$, for whatever choice of M, if we make u sufficiently small. Hence the Lipschitz condition fails.

The minimum and maximum are conjunctive and disjunctive extended aggregation functions respectively, and simultaneously limiting cases of averaging aggregation functions.

Both minimum and maximum are symmetric and associative, and Lipschitz-continuous (in fact kernel aggregation functions). The min function has the neutral element $e = 1$ and the absorbing element $a = 0$, and the max function has the neutral element $e = 0$ and the absorbing element $a = 1$. They are dual to each other with respect to the standard negation $N(t) = 1 - t$ (and in fact, any strong negation N)

$$\max(\mathbf{x}) = 1 - \min(\mathbf{1} - \mathbf{x}) = 1 - \min_{i=1,\dots,n}(1 - x_i),$$

$$\max(\mathbf{x}) = N(\min(N(\mathbf{x})) = N(\min_{i=1,\dots,n}(N(x_i)),$$

Most classes and parametric families of aggregation functions include maximum and minimum as members or as the limiting cases.

1.4.2 Means

Means are averaging aggregation functions. Formally, a mean is simply a function f with the property [40]

$$\min(\mathbf{x}) \le f(\mathbf{x}) \le \max(\mathbf{x}).$$

Still there are other properties that define one or another family of means. We discuss them in Chapter 2.

Definition 1.65 (Arithmetic mean). *The arithmetic mean is the function*

$$M(\mathbf{x}) = \frac{1}{n}(x_1 + x_2 + \ldots + x_n) = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

Definition 1.66 (Weighting vector). *A vector* $\mathbf{w} = (w_1, \ldots, w_n)$ *is called a weighting vector if* $w_i \in [0, 1]$ *and* $\sum\limits_{i=1}^{n} w_i = 1$.

Definition 1.67 (Weighted arithmetic mean). *Given a weighting vector* $\mathbf{w}$*, the weighted arithmetic mean is the function*

$$M_{\mathbf{w}}(\mathbf{x}) = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n = \sum_{i=1}^{n} w_i x_i.$$

Definition 1.68 (Geometric mean). *The geometric mean is the function*

$$G(\mathbf{x}) = \sqrt[n]{x_1 x_2 \dots x_n} = \left(\prod_{i=1}^{n} x_i\right)^{1/n}.$$

Definition 1.69 (Harmonic mean). *The harmonic mean is the function*

$$H(\mathbf{x}) = n\left(\sum_{i=1}^{n} \frac{1}{x_i}\right)^{-1}.$$

We discuss weighted versions of the above means, as well as many other means in Chapter 2.

1.4.3 Medians

Definition 1.70 (Median). *The median is the function*

$$Med(\mathbf{x}) = \begin{cases} \frac{1}{2}(x_{(k)} + x_{(k+1)}), & \text{if } n = 2k \text{ is even} \\ x_{(k)}, & \text{if } n = 2k-1 \text{ is odd,} \end{cases}$$

where $x_{(k)}$ is the k-th largest (or smallest) component of $\mathbf{x}$.

Definition 1.71 (a-Median). *Given a value $a \in [0,1]$, the a-median is the function*

$$Med_a(\mathbf{x}) = Med(x_1, \dots, x_n, \overbrace{a, \dots, a}^{n-1 \text{ times}}).$$

1.4.4 Ordered weighted averaging

Ordered weighted averaging functions (OWA) are also averaging aggregation functions, which associate weights not with a particular input, but rather with its value. They have been introduced by Yager [263] and have become very popular in the fuzzy sets community.

Let $\mathbf{x}_{\searrow}$ be the vector obtained from $\mathbf{x}$ by arranging its components in non-increasing order $x_{(1)} \geq x_{(2)} \geq \dots \geq x_{(n)}$.

Definition 1.72 (OWA). *Given a weighting vector* $\mathbf{w}$*, the OWA function is*

$$OWA_{\mathbf{w}}(\mathbf{x}) = \sum_{i=1}^{n} w_i x_{(i)} =< \mathbf{w}, \mathbf{x}_{\searrow} > .$$

Note that calculation of the value of the OWA function can be done by using a `sort()` operation. If all weights are equal, OWA becomes the arithmetic mean. The vector of weights $\mathbf{w} = (1, 0, \ldots, 0)$ yields the maximum and $\mathbf{w} = (0, \ldots, 0, 1)$ yields the minimum function.

1.4.5 Choquet and Sugeno integrals

These are two classes of averaging aggregation functions defined with respect to a fuzzy measure. They are useful to model interactions between the variables x_i.

Definition 1.73 (Fuzzy measure). *Let* $\mathcal{N} = \{1, 2, \ldots, n\}$*. A discrete fuzzy measure is a set function*[19] $v : 2^{\mathcal{N}} \to [0, 1]$ *which is monotonic (i.e.* $v(S) \leq v(T)$ *whenever* $S \subseteq T$*) and satisfies* $v(\emptyset) = 0, v(\mathcal{N}) = 1$*.*

Definition 1.74 (Choquet integral). *The discrete Choquet integral with respect to a fuzzy measure* v *is given by*

$$C_v(\mathbf{x}) = \sum_{i=1}^{n} x_{(i)} [v(\{j | x_j \geq x_{(i)}\}) - v(\{j | x_j \geq x_{(i+1)}\})], \tag{1.9}$$

where $\mathbf{x}_{\nearrow} = (x_{(1)}, x_{(2)}, \ldots, x_{(n)})$ *is a non-decreasing permutation of the input* $\mathbf{x}$*, and* $x_{(n+1)} = \infty$ *by convention.*

By rearranging the terms of the sum, Eq. (1.9) can also be written as

$$C_v(\mathbf{x}) = \sum_{i=1}^{n} \left[x_{(i)} - x_{(i-1)} \right] v(H_i). \tag{1.10}$$

where $x_{(0)} = 0$ by convention, and $H_i = \{(i), \ldots, (n)\}$ is the subset of indices of $n - i + 1$ largest components of $\mathbf{x}$.

The class of Choquet integrals includes weighted arithmetic means and OWA functions as special cases. The Choquet integral is a piecewise linear idempotent function, uniquely defined by its values at the vertices of the unit cube $[0, 1]^n$, i.e., at the points $\mathbf{x}$, whose coordinates $x_i \in \{0, 1\}$. Note that there are 2^n such points, the same as the number of values that determine the fuzzy measure v. We consider these functions in detail in Chapter 2.

[19] A set function is a function whose domain consists of all possible subsets of $\mathcal{N}$. For example, for $n = 3$, a set function is specified by $2^3 = 8$ values at $v(\emptyset)$, $v(\{1\})$, $v(\{2\})$, $v(\{3\})$, $v(\{1, 2\})$, $v(\{1, 3\})$, $v(\{2, 3\})$, $v(\{1, 2, 3\})$.

Definition 1.75 (Sugeno integral). *The Sugeno integral with respect to a fuzzy measure v is given by*

$$S_v(\mathbf{x}) = \max_{i=1,\dots,n} \min\{x_{(i)}, v(H_i)\}, \tag{1.11}$$

where $\mathbf{x}_{\nearrow} = (x_{(1)}, x_{(2)}, \dots, x_{(n)})$ is a non-decreasing permutation of the input $\mathbf{x}$, and $H_i = \{(i), \dots, (n)\}$.

In the special case of a symmetric fuzzy measure (i.e., when $v(H_i) = v(|H_i|)$ depends only on the cardinality of the set H_i), Sugeno integral becomes the median $S_v(\mathbf{x}) = Med(x_1, \dots, x_n, 1, v(n-1), v(n-2), \dots, v(1))$.

1.4.6 Conjunctive and disjunctive functions

The prototypical examples of conjunctive and disjunctive aggregation functions are so-called triangular norms and conorms respectively (t-norms and t-conorms). They are treated in detail in Chapter 3, and below are just a few typical examples. All functions in Examples 1.76-1.79 are symmetric and associative.

Example 1.76. The product is a conjunctive extended aggregation function (it is a t–norm)

$$T_P(\mathbf{x}) = \prod_{i=1}^{n} x_i.$$

Example 1.77. The dual product, also called probabilistic sum, is a disjunctive extended aggregation function (it is a t–conorm)

$$S_P(\mathbf{x}) = 1 - \prod_{i=1}^{n} (1 - x_i).$$

Example 1.78. Łukasiewicz triangular norm and conorm are conjunctive and disjunctive extended aggregation functions

$$T_L(\mathbf{x}) = \max(0, \sum_{i=1}^{n} x_i - (n-1)),$$

$$S_L(\mathbf{x}) = \min(1, \sum_{i=1}^{n} x_i).$$

Example 1.79. Einstein sum is a disjunctive aggregation function (it is a t–conorm). Its bivariate form is given by

$$f(x_1, x_2) = \frac{x_1 + x_2}{1 + x_1 x_2}.$$

Example 1.80. The function

$$f(x_1, x_2) = x_1 x_2^2$$

is a conjunctive ($x_1 x_2^2 \leq x_1 x_2 \leq \min(x_1, x_2)$), asymmetric aggregation function. It is not a t-norm.

1.4.7 Mixed aggregation

In some situations, high input values are required to reinforce each other whereas low values pull the output down. Thus the aggregation function has to be disjunctive for high values, conjunctive for low values, and perhaps averaging if some values are high and some are low. This is typically the case when high values are interpreted as "positive" information, and low values as "negative" information. The classical expert systems MYCIN and PROSPECTOR [38, 88] use precisely this type of aggregation (on [-1,1] interval).

A different behavior may also be needed: aggregation of both high and low values moves the output towards some intermediate value. Thus certain aggregation functions need to be conjunctive, disjunctive or averaging in different parts of their domain.

Uninorms and nullnorms (see Chapter 4) are typical examples of such aggregation functions, but there are many others.

Example 1.81. The $3 - \Pi$ function [278] is

$$f(\mathbf{x}) = \frac{\prod_{i=1}^{n} x_i}{\prod_{i=1}^{n} x_i + \prod_{i=1}^{n} (1 - x_i)},$$

with the convention $\frac{0}{0} = 0$. It is conjunctive on $[0, \frac{1}{2}]^n$, disjunctive on $[\frac{1}{2}, 1]^n$ and averaging elsewhere. It is associative, with the neutral element $e = \frac{1}{2}$, and discontinuous on the boundaries of $[0, 1]^n$. It is a uninorm.

1.5 Composition and transformation of aggregation functions

We have examined several prototypical examples of aggregation functions from different classes. Of course, this is a very limited number of functions, and they may not be sufficient to model a specific problem. The question arises as to how we can construct new aggregation functions from the existing ones. Which properties will be conserved, and which properties will be lost?

We consider two simple techniques for constructing new aggregation functions. The first technique is based on the monotonic transformation of the

inputs and the second is based on iterative application of aggregation functions.

Let us consider univariate strictly increasing bijections (hence continuous) $\varphi_1, \varphi_2, \ldots, \varphi_n$ and ψ; $\varphi_i, \psi : [0,1] \to [0,1]$.

Proposition 1.82. *Let $\varphi_1, \ldots, \varphi_n, \psi : [0,1] \to [0,1]$ be strictly increasing bijections. For any aggregation function f, the function*

$$g(\mathbf{x}) = \psi(f(\varphi_1(x_1), \varphi_2(x_2), \ldots, \varphi_n(x_n)))$$

is an aggregation function.

Note 1.83. The functions φ_i, ψ may also be strictly decreasing (but all at the same time), we already saw in section 1.3.3 that if we choose each φ_i and ψ as a strong negation, then we obtain a dual aggregation function g.

Of course, nothing can be said about the properties of g. However in some special cases we can establish which properties remain intact.

Proposition 1.84. *Let f be an aggregation function and let g be defined as in Proposition 1.82. Then*

- *If f is continuous, so is g.*
- *If f is symmetric and $\varphi_1 = \varphi_2 = \ldots = \varphi_n$, then g is symmetric.*
- *If f is associative and $\psi^{-1} = \varphi_1 = \ldots = \varphi_n$ then g is associative.*

Next, take $\varphi_1(t) = \varphi_2(t) = \ldots = \varphi_n(t) = t$. That is, consider a composition of functions $g(\mathbf{x}) = (\psi \circ f)(\mathbf{x}) = \psi(f(\mathbf{x}))$. We examine how the choice of ψ can affect the behavior of the aggregation function f.

It is clear that ψ needs to be monotone non-decreasing. Depending on its properties, it can modify the type of the aggregation function.

Proposition 1.85. *Let f be an aggregation function and let $\psi : [0,1] \to [0,1]$ be a non-decreasing function satisfying $\psi(0) = 0$ and $\psi(1) = 1$. If $\psi(f(1, \ldots, 1, t, 1, \ldots, 1)) \leq t$ for all $t \in [0,1]$ and at any position, then $g = \psi \circ f$ is a conjunctive aggregation function.*

Proof. The proof is simple: For any fixed position i, and any $\mathbf{x}$ we have $g(\mathbf{x}) \leq \max\limits_{x_j \in [0,1], j \neq i} g(\mathbf{x}) = g(1, \ldots, 1, x_i, 1 \ldots, 1) \leq x_i$. This holds for every i, therefore $g(\mathbf{x}) \leq \min(\mathbf{x})$. By applying Proposition 1.82 we complete the proof.

Proposition 1.85 will be mainly used when choosing f as an averaging aggregation function. Not every averaging aggregation function can be converted to a conjunctive function using Proposition 1.85: the value $f(\mathbf{x})$ must be distinct from 1 for $\mathbf{x} \neq (1, 1, \ldots, 1)$. Furthermore, if $\max\limits_{i=1,\ldots,n} f(1, \ldots, 1, 0, 1, \ldots, 1)$ $= a < 1$ with 0 in the i-th position, then necessarily $\psi(t) = 0$ for $t \leq a$. If ψ is a bijection, then f must have $a = 0$ as absorbing element. The main point

of Proposition 1.85 is that one can construct conjunctive aggregation functions from many types of averaging functions (discussed in Chapter 2) by a simple transformation, and that its condition involves single variate functions $\psi(f(1,\ldots,1,t,1,\ldots,1))$, which is not difficult to verify. We will use this simple result in Section 3.4.16 when constructing asymmetric conjunctive and disjunctive functions.

Note 1.86. A similar construction also transforms averaging functions with the absorbing element $a = 1$, if ψ is a bijection, to disjunctive functions (by using duality). However it does not work the other way around, i.e., to construct averaging functions from either conjunctive or disjunctive functions. This can be achieved by using the idempotization method, see [43], p.28.

Example 1.87. Take the geometric mean $f(\mathbf{x}) = \sqrt{x_1 x_2}$, which is an averaging function with the absorbing element $a = 0$. Take $\psi(t) = t^2$. Composition $(\psi \circ f)(\mathbf{x}) = x_1 x_2$, yields the product function, which is conjunctive.

Example 1.88. Take the harmonic mean $f(\mathbf{x}) = 2(\frac{1}{x_1} + \frac{1}{x_2})^{-1} = \frac{2x_1x_2}{x_1+x_2}$, which also has the absorbing element $a = 0$. Take again $\psi(t) = t^2$. Composition $g(\mathbf{x}) = (\psi \circ f)(\mathbf{x}) = \frac{(2x_1x_2)^2}{(x_1+x_2)^2}$ is a conjunctive aggregation function (we can check that $g(x_1, 1) = \frac{4x_1^2}{(1+x_1)^2} \le x_1$). Now take $\psi(t) = \frac{t}{2-t}$. A simple computation yields $g(x_1, 1) = x_1$ and $g(x_1, x_2) = \frac{x_1x_2}{x_1+x_2-x_1x_2}$, a Hamacher triangular norm T_0^H, see p.152 which is conjunctive.

Let us now consider an iterative application of aggregation functions. Consider three aggregation functions $f, g : [0,1]^n \to [0,1]$ and $h : [0,1]^2 \to [0,1]$, i.e., h is a bivariate function. Then the combination

$$H(\mathbf{x}) = h(f(\mathbf{x}), g(\mathbf{x}))$$

is also an aggregation function. It is continuous if f, g and h are. Depending on the properties of these functions, the resulting aggregation function may also possess certain properties.

Proposition 1.89. *Let f and g be n-ary aggregation functions, h be a bivariate aggregation function, and let H be defined as $H(\mathbf{x}) = h(f(\mathbf{x}), g(\mathbf{x}))$. Then*

- *If f and g are symmetric then H is also symmetric.*
- *If f, g and h are averaging functions, then H is averaging.*
- *If f, g and h are associative, H is not necessarily associative.*
- *If any or all f, g and h have a neutral element, H does not necessarily have a neutral element.*
- *If f, g and h are conjunctive (disjunctive), H is also conjunctive (disjunctive).*

Previously we mentioned that in certain applications the use of $[0,1]$ scale is not very intuitive. One situation is when we aggregate pieces of "positive" and "negative" information, for instance evidence that confirms and disconfirms a hypothesis. It may be more natural to use a bipolar $[-1,1]$ scale, in which negative values refer to negative evidence and positive values refer to positive evidence. In some early expert systems (MYCIN [38] and PROSPECTOR [88]) the $[-1,1]$ scale was used.

The question is whether the use of a different scale brings anything new to the mathematics of aggregation. The answer is negative, the aggregation functions on two different closed intervals are isomorphic, i.e., any aggregation function on the scale $[a,b]$ can be obtained by a simple linear transformation from an aggregation function on $[0,1]$. Thus, the choice of the scale is a question of interpretability, not of the type of aggregation.

Transformation from one scale to another is straightforward, and it can be done in many different ways. The most common formulas are the following. Let $f^{[a,b]}$ be an aggregation function on the interval $[a,b]$, and let $f^{[0,1]}$ be the corresponding aggregation function on $[0,1]$. Then

$$f^{[a,b]}(x_1,\ldots,x_n) = (b-a)f^{[0,1]}\left(\frac{x_1-a}{b-a},\ldots,\frac{x_n-a}{b-a}\right)+a, \tag{1.12}$$

$$f^{[0,1]}(x_1,\ldots,x_n) = \frac{f^{[a,b]}\left((b-a)x_1+a,\ldots,(b-a)x_n+a\right)-a}{b-a}, \tag{1.13}$$

or in vector form

$$f^{[a,b]}(\mathbf{x}) = (b-a)f^{[0,1]}\left(\frac{\mathbf{x}-\mathbf{a}}{b-a}\right)+a,$$

$$f^{[0,1]}(\mathbf{x}) = \frac{f^{[a,b]}\left((b-a)(\mathbf{x}+\mathbf{a})\right)-a}{b-a}.$$

Thus for transformation to and from a bipolar scale we use

$$f^{[-1,1]}(x_1,\ldots,x_n) = 2f^{[0,1]}\left(\frac{x_1+1}{2},\ldots,\frac{x_n+1}{2}\right)-1, \tag{1.14}$$

$$f^{[0,1]}(x_1,\ldots,x_n) = \frac{f^{[-1,1]}\left(2x_1-1,\ldots,2x_n-1\right)+1}{2}. \tag{1.15}$$

1.6 How to choose an aggregation function

There are infinitely many aggregation functions. They are grouped in various families, such as means, triangular norms and conorms, Choquet and Sugeno integrals, and many others. The question is how to choose the most suitable aggregation function for a specific application. Is one aggregation function

enough, or should different aggregation functions be used in different parts of the application?

There are two components to the answer. First of all, the selected aggregation function must be consistent with the semantics of the aggregation procedure. That is, if one models a conjunction, averaging or disjunctive aggregation functions are not suitable. Should the aggregation function be symmetric, have a neutral or absorbing element, or be idempotent? Is the number of inputs always the same? What is the interpretation of input values? Answering these questions should result in a number of mathematical properties, based on which a suitable class or family can be chosen.

The second issue is to choose the appropriate member of that class or family, which does what it is supposed to do — produces adequate outputs for given inputs. It is expected that the developer of a system has some rough idea of what the appropriate outputs are for some prototype inputs. Thus we arrive at the issue of fitting the data.

The data may come from different sources and in different forms. First, it could be the result of some mental experiment: let us take the input values $(1, 0, 0)$. What output do we expect?

Second, the developer of an application could ask the domain experts to provide their opinion on the desired outputs for selected inputs. This can be done by presenting the experts some prototypical cases (either the input vectors, or domain specific situations before they are translated into the inputs). If there is more than one expert, their outputs could be either averaged, or translated into the range of possible output values, or the experts could be brought together to find a consensus.

Third, the data could be collected in an experiment, by asking a group of lay people or experts about their input and output values, but without associating these values with some aggregation rule. For example, an interesting experiment reported in [286, 287] consisted in asking a group of people about the membership values they would use for different objects in the fuzzy sets "metallic", "container", and then in the combined set "metallic container". The goal was to determine a model for intersection of two sets. The subjects were asked the questions about membership values on three separate days, to discourage them from building some inner model for aggregation.

Fourth, the data can be collected automatically by observing the responses of subjects to various stimuli. For example, by presenting a user of a computer system with some information and recording their actions or decisions.

In the most typical case, the data comes in pairs $(\mathbf{x}, y)$, where $\mathbf{x} \in [0, 1]^n$ is the input vector and $y \in [0, 1]$ is the desired output. There are several pairs, which will be denoted by a subscript k: $(\mathbf{x}_k, y_k), k = 1, \ldots, K$. However there are variations of the data set: a) some components of vectors $\mathbf{x}_k$ may be missing, b) vectors $\mathbf{x}_k$ may have varying dimension by construction, and c) the outputs y_k could be specified as a range of values (i.e., the interval $[\underline{y}_k, \overline{y}_k]$).

In fitting an aggregation function to the data, we will distinguish interpolation and approximation problems. In the case of interpolation, our aim is to fit the specified output values exactly. For instance, the pairs $((0, 0, \ldots, 0), 0)$ and $((1, 1, \ldots, 1), 1)$ should always be interpolated. On the other hand, when the data comes from an experiment, it will normally contain some errors, and therefore it is pointless to interpolate the inaccurate values y_k. In this case our aim is to stay close to the desired outputs without actually matching them. This is the approximation problem.

There are of course other issues to take into account when choosing an aggregation function, such as simplicity, numerical efficiency, easiness of interpretation, and so on [286]. There are no general rules here, and it is up to the system developer to make an educated choice. In what follows, we concentrate on the first two criteria: to be consistent with semantically important properties of the aggregation procedure, and to fit the desired data.

We now formalize the selection problem.

Problem 1.90 (Selection of an aggregation function). Let us have a number of mathematical properties $\mathcal{P}_1, \mathcal{P}_2, \ldots$ and the data $\mathcal{D} = \{(\mathbf{x}_k, y_k)\}_{k=1}^K$. Choose an aggregation function f consistent with $\mathcal{P}_1, \mathcal{P}_2, \ldots$, and satisfying $f(\mathbf{x}_k) \approx y_k, k = 1, \ldots, K$.

The approximate equalities may of course be satisfied exactly, if the properties $\mathcal{P}_1, \mathcal{P}_2, \ldots$ allow this. We shall also consider a variation of the selection problem when y_k are given as intervals, in which case we require $f(\mathbf{x}_k) \in [\underline{y}_k, \overline{y}_k]$, or even approximately satisfy this condition.

The satisfaction of approximate equalities $f(\mathbf{x}_k) \approx y_k$ is usually translated into the following minimization problem.

$$\begin{aligned} &\text{minimize } ||\mathbf{r}|| \\ &\text{subject to } f \text{ satisfies } \ \mathcal{P}_1, \mathcal{P}_2, \ldots, \end{aligned} \tag{1.16}$$

where $||\mathbf{r}||$ is the norm of the residuals, i.e., $\mathbf{r} \in R^K$ is the vector of the differences between the predicted and observed values $r_k = f(\mathbf{x}_k) - y_k$. There are many ways to choose the norm, and the most popular are the least squares norm

$$||\mathbf{r}||_2 = \left(\sum_{k=1}^K r_k^2 \right)^{1/2},$$

the least absolute deviation norm

$$||\mathbf{r}||_1 = \sum_{k=1}^K |r_k|,$$

the Chebyshev norm

$$||\mathbf{r}||_\infty = \max_{k=1,\ldots,K} |r_k|,$$

or their weighted analogues, like

$$||\mathbf{r}|| = \left(\sum_{k=1}^{K} u_k r_k^2\right)^{1/2},$$

where the weight $u_k \geq 0$ determines the relative importance to fit the k-th value y_k [20].

Example 1.91. Consider choosing the weights of a weighted arithmetic mean consistent with the data set $\{(\mathbf{x}_k, y_k)\}_{k=1}^K$ using the least squares approach. We minimize the sum of squares

$$\begin{array}{ll} \text{minimize} & \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i x_{ik} - y_k\right)^2 \\ \text{subject to} & \sum_{i=1}^{n} w_i = 1, \\ & w_1, \ldots, w_n \geq 0. \end{array}$$

This is a quadratic programming problem (see Section A.5), which is solved by a number of standard methods.

In some studies [138] it was suggested that for decision making problems, the actual numerical value of the output $f(\mathbf{x}_k)$ was not as important as the ranking of the outputs. For instance, if $y_k \leq y_l$, then it should be $f(\mathbf{x}_k) \leq f(\mathbf{x}_l)$. Indeed, people are not really good at assigning consistent numerical scores to their preferences, but they are good at ranking the alternatives. Thus it is argued [138] that a suitable choice of aggregation function should be consistent with the ranking of the outputs y_k rather than their numerical values. The use of the mentioned fitting criteria does not preserve the ranking of outputs, unless they are interpolated. Preservation of ranking of outputs can be done by imposing the constraints $f(\mathbf{x}_k) \leq f(\mathbf{x}_l)$ if $y_k \leq y_l$ for all pairs k, l. We will consider this in detail in Chapter 5.

1.7 Numerical approximation and optimization tools

The choice of aggregation functions based on the empirical data requires solving a *regression* (or *approximation*) problem, subject to certain constraints, see Problem (1.16). In this section we briefly outline a number of useful numerical tools that will allow us to solve such problems. A more detailed discussion is provided in the Appendix A, p.305.

An approximation problem involves fitting a function from a certain class to the data $(\mathbf{x}_k, y_k), k = 1, \ldots, K$. If the data can be fitted exactly, it is

[20] Values y_k may have been recorded with different accuracies, or specified by experts of different standing.

called an *interpolation* problem. The goal of approximation/interpolation is to build a model of the function f, which allows one to calculate its values at $\mathbf{x}$ different from the data. There are many different methods of approximation, starting from linear regression, polynomial regression, spline functions, radial basis functions, neural networks and so on.

The best studied case is the univariate approximation/interpolation. The methods of Newton and Lagrange interpolation, polynomial and spline approximation are the classical tools. However it is often needed to impose additional restrictions on the approximation f, the most important for us will be monotonicity. The methods of *constrained* approximation discussed in the Appendix A will be often used for construction of some classes of aggregation functions.

Of course, aggregation functions are multivariate functions, hence the methods of univariate approximation are generally not applicable. The data do not have a special structure, so we will need methods of multivariate *scattered* data approximation. Further, aggregation functions have additional properties (at the very least monotonicity, but other properties like idempotency, symmetry and neutral element may be needed). Hence we need methods of constrained multivariate approximation. Specifically we will employ methods of tensor-product spline approximation and Lipschitz approximation, outlined in the Appendix A.

An approximation problem typically involves a solution to an optimization problem of type (1.16). Depending on the properties of the chosen norm $||\mathbf{r}||$ and on the properties $\mathcal{P}_i$, it may turn out to be a general nonlinear optimization problem, or a problem from a special class. The latter is very useful, as certain optimization problems have well researched solution techniques and proven algorithms. It is important to realize this basic fact, and to formulate the approximation problem in such a way that its special structure can be exploited, rather than attempting to solve it with raw computational power.

It is also important to realize that a general non-linear optimization problem typically has many local optima, which are not the actual solutions (i.e., not the absolute minima and maxima). This is illustrated on Fig. A.4 on p. 323. The majority of methods of non-linear minimization are *local methods*, i.e., they converge to a local optimum of the objective function, not to its absolute (or global) minimum [21].

The number of local optima could be very high, of order of $10^{10} - 10^{60}$, it grows exponentially with the number of variables. Therefore their explicit enumeration is practically infeasible. There are several *global optimization* methods for this type of problem (examples are random start, simulated annealing, tabu search, genetic algorithms, deterministic optimization), which we discuss in the Appendix A.

[21] Frequently the term *globally convergent* is used to characterize such local methods. It means that the method converges to a *local* minimum from *any* initial approximation, not that it converges to the global minimum.

On the other hand, some structured optimization problems involve a convex objective function (or a variant of it). In such cases there is a unique local minimum, which is therefore the global minimum. Local optimization methods will easily find it. The difficulty in this kind of problem typically lies in the constraints. When the constraints involve linear equalities and inequalities, and the objective function is either linear or convex quadratic, the problem is called either a linear or quadratic programming problem (LP or QP). These two special problems are extensively studied, and a number of very efficient algorithms for their solution are available (see Appendix A).

We want to stress the need to apply the right tool for solving each type of approximation or optimization problem. Generic off-the-shelf methods would be extremely inefficient if one fails to recognize and properly deal with a special structure of the problem. Even if the optimization problem is linear or convex in just a few variables, it is extremely helpful to identify those variables and apply efficient specialized methods for solving the respective sub-problems.

In this book we will extensively rely on solving the following problems.

- Constrained least squares regression (includes non-negative least squares);
- Constrained least absolute deviation regression;
- Spline interpolation and approximation;
- Multivariate monotone interpolation and approximation;
- Linear programming;
- Quadratic programming;
- Unconstrained non-linear programming;
- Convex optimization;
- Univariate and multivariate global optimization.

If the reader is not familiar with the techniques for solving the mentioned problems, a brief description of each problem and the available tools for their solution is given in Appendix A. It also contains references to the current implementations of the mentioned algorithms.

1.8 Key references

B. Bouchon-Meunier, editor, *Aggregation and Fusion of Imperfect Information*, Physica-Verlag, Heidelberg, 1998.

T. Calvo, G. Mayor, and R. Mesiar, editors, *Aggregation Operators. New Trends and Applications.* Physica-Verlag, Heidelberg, New York, 2002.

D. Dubois and H. Prade. *Fuzzy Sets and Systems. Theory and Applications.* Academic Press, New York, 1980.

D. Dubois and H. Prade. A review of fuzzy set aggregation connectives. *Inform. Sci.*, 36:85–121, 1985.

D. Dubois and H. Prade. *Fundamentals of Fuzzy Sets.* Kluwer, Boston, 2000.

D. Dubois and H. Prade. On the use of aggregation operations in information fusion processes. *Fuzzy Sets and Systems*, 142:143–161, 2004.

J. Fodor and M. Roubens. *Fuzzy Preference Modelling and Multicriteria Decision Support.* Kluwer, Dordrecht, 1994.

M. Grabisch, H.T. Nguyen, and E.A. Walker. *Fundamentals of Uncertainty Calculi, with Applications to Fuzzy Inference.* Kluwer, Dordrecht, 1995.

E.P. Klement, R. Mesiar, and E. Pap. *Triangular Norms.* Kluwer, Dordrecht, 2000.

R. Yager and D. Filev. *Essentials of Fuzzy Modelling and Control.* J. Wiley & Sons, New York, 1994.

H.-J. Zimmermann. *Fuzzy Set Theory - and its Applications.* Kluwer, Boston, 1996.

2

Averaging Functions

2.1 Semantics

Averaging is the most common way to combine inputs. It is commonly used in voting, multicriteria and group decision making, constructing various performance scores, statistical analysis, etc. The basic rule is that the total score cannot be above or below any of the inputs. The aggregated value is seen as some sort of representative value of all the inputs.

We shall adopt the following generic definition [40, 43, 83, 87].

Definition 2.1 (Averaging aggregation). *An aggregation function f is averaging if for every $\mathbf{x}$ it is bounded by*

$$\min(\mathbf{x}) \leq f(\mathbf{x}) \leq \max(\mathbf{x}).$$

We remind that due to monotonicity of aggregation functions, averaging functions are idempotent, and vice versa, see Note 1.12, p. 9. That is, an aggregation function f is averaging if and only if $f(t, \ldots, t) = t$ for any $t \in [0, 1]$.

Formally, the minimum and maximum functions can be considered as averaging, however they are the limiting cases, right on the border with conjunctive and disjunctive functions, and will be treated in Chapter 3. There are also some types of mixed aggregation functions, such as uninorms or nullnorms, that include averaging functions as particular cases; these will be treated in Chapter 4.

Measure of orness

The *measure of orness*, also called the *degree of orness* or *attitudinal character*, is an important numerical characteristic of averaging aggregation functions. It was first defined in 1974 by Dujmovic [89, 90], and then rediscovered several times, see [97, 263], mainly in the context of OWA functions (Section 2.5). It

is applicable to any averaging function (and even to some other aggregation functions, like ST-OWA [92], see Chapter 4).

Basically, the measure of orness measures how far a given averaging function is from the max function, which is the weakest disjunctive function. The measure of orness is computed for any averaging function [90, 92, 97] using

---

Definition 2.2 (Measure of orness). *Let f be an averaging aggregation function. Then its measure of orness is*

$$orness(f) = \frac{\int_{[0,1]^n} f(\mathbf{x})d\mathbf{x} - \int_{[0,1]^n} \min(\mathbf{x})d\mathbf{x}}{\int_{[0,1]^n} \max(\mathbf{x})d\mathbf{x} - \int_{[0,1]^n} \min(\mathbf{x})d\mathbf{x}}. \tag{2.1}$$

Clearly, $orness(\max) = 1$ and $orness(\min) = 0$, and for any f, $orness(f) \in [0,1]$. The calculation of the integrals of max and min functions was performed in [89] and results in simple equations

$$\int_{[0,1]^n} \max(\mathbf{x})d\mathbf{x} = \frac{n}{n+1} \text{ and } \int_{[0,1]^n} \min(\mathbf{x})d\mathbf{x} = \frac{1}{n+1}. \tag{2.2}$$

A different measure of orness, the average orness value, is proposed in [92].

Definition 2.3 (Average orness value). *Let f be an averaging aggregation function. Then its average orness value is*

$$\overline{orness}(f) = \int_{[0,1]^n} \frac{f(\mathbf{x}) - \min(\mathbf{x})}{\max(\mathbf{x}) - \min(\mathbf{x})} d\mathbf{x}. \tag{2.3}$$

Both the measure of orness and the average orness value are $\frac{1}{2}$ for weighted arithmetic means, and later we will see that both quantities coincide for OWA functions. However computation of the average orness value for other averaging functions is more involved (typically performed by numerical methods), therefore we will use mainly the measure of orness in Definition 2.2.

2.2 Classical means

Means are often treated synonymously with averaging functions. However, the classical treatment of means (see, e.g., [40]) excludes certain types of averaging functions, which have been developed quite recently, in particular ordered weighted averaging and various integrals. On the other hand some classical means (e.g., some Gini means) lack monotonicity, and therefore are not aggregation functions. Following the tradition, in this section we will concentrate on various classical means, and present other types of averaging, or mean-type functions in separate sections.

Arithmetic mean

The arithmetic mean is the most widely used aggregation function.

Definition 2.4 (Arithmetic mean). *The arithmetic mean, or the average of n values, is the function*

$$M(\mathbf{x}) = \frac{1}{n}(x_1 + x_2 + \ldots + x_n) = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

Since M is properly defined for any number of arguments, it is an extended aggregation function, see Definition 1.6.

Main properties

- The arithmetic mean M is a strictly increasing aggregation function;
- M is a symmetric function;
- M is an additive function, i.e., $M(\mathbf{x}+\mathbf{y}) = M(\mathbf{x}) + M(\mathbf{y})$ for all $\mathbf{x}, \mathbf{y} \in [0,1]^n$ such that $\mathbf{x}+\mathbf{y} \in [0,1]^n$;
- M is a homogeneous function, i.e., $M(\lambda\mathbf{x}) = \lambda M(\mathbf{x})$ for all $\mathbf{x} \in [0,1]^n$ and for all $\lambda \in [0,1]$;
- The orness measure $orness(M) = \frac{1}{2}$;
- M is a Lipschitz continuous function, with the Lipschitz constant in any $||\cdot||_p$ norm (see p. 22) $n^{-1/p}$, the smallest Lipschitz constant of all aggregation functions.

When the inputs are not symmetric, it is a common practice to associate each input with a weight, a number $w_i \in [0,1]$ which reflects the relative contribution of this input to the total score. For example, in shareholders' meetings, the strength of each vote is associated with the number of shares this shareholder possesses. The votes are usually just added to each other, and after dividing by the total number of shares, we obtain a weighted arithmetic mean. Weights can also represent the reliability of an input or its importance.

Weights are not the only way to obtain asymmetric functions, we will study other methods in Section 2.4 and in Chapters 3 and 4. Recall from Chapter 1 the definition of a weighting vector:

Definition 2.5 (Weighting vector). *A vector* $\mathbf{w} = (w_1, \ldots, w_n)$ *is called a weighting vector if* $w_i \in [0,1]$ *and* $\sum\limits_{i=1}^{n} w_i = 1$.

Definition 2.6 (Weighted arithmetic mean). *Given a weighting vector* $\mathbf{w}$*, the weighted arithmetic mean is the function*

$$M_{\mathbf{w}}(\mathbf{x}) = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n = \sum_{i=1}^{n} w_i x_i = < \mathbf{w}, \mathbf{x} > .$$

Main properties

- A weighted arithmetic mean $M_{\mathbf{w}}$ is a strictly increasing aggregation function, if all $w_i > 0$;
- $M_{\mathbf{w}}$ is an asymmetric (unless $w_i = 1/n$ for all $i \in \{1, \ldots, n\}$) idempotent function;
- $M_{\mathbf{w}}$ is an additive function, i.e., $M_{\mathbf{w}}(\mathbf{x} + \mathbf{y}) = M_{\mathbf{w}}(\mathbf{x}) + M_{\mathbf{w}}(\mathbf{y})$ for all $\mathbf{x}, \mathbf{y} \in [0,1]^n$ such that $\mathbf{x} + \mathbf{y} \in [0,1]^n$;
- $M_{\mathbf{w}}$ is a homogeneous function, i.e., $M_{\mathbf{w}}(\lambda \mathbf{x}) = \lambda M_{\mathbf{w}}(\mathbf{x})$ for all $\mathbf{x} \in [0,1]^n$ and for all $\lambda \in [0,1]$;
- Jensen's inequality: for any convex function[1] $g : [0,1] \to [-\infty, \infty]$, $g(M_{\mathbf{w}}(\mathbf{x})) \leq M_{\mathbf{w}}(g(x_1), \ldots, g(x_n))$.
- $M_{\mathbf{w}}$ is a Lipschitz continuous function, in fact it is a kernel aggregation function (see p. 23);
- $M_{\mathbf{w}}$ is a shift-invariant function (see p. 17) ;
- The orness measure $orness(M_{\mathbf{w}}) = \frac{1}{2}$; [2]
- $M_{\mathbf{w}}$ is a special case of the Choquet integral (see Section 2.6) with respect to an additive fuzzy measure.

Geometric and harmonic means

Weighted arithmetic means are good for averaging inputs that can be added together. Frequently the inputs are not added but multiplied. For example, when averaging the rates of investment return over several years the use of the arithmetic mean is incorrect. This is because the rate of return (say 10%) signifies that in one year the investment was multiplied by a factor 1.1. If the return is 20% in the next year, then the total is multiplied by 1.2, which means that the original investment is multiplied by a factor of 1.1×1.2. The average return is calculated by using the geometric mean of 1.1 and 1.2, which gives ≈ 1.15.

[1] A function g is convex if and only if $g(\alpha t_1 + (1-\alpha)t_2) \leq \alpha g(t_1) + (1-\alpha) g(t_2)$ for all $t_1, t_2 \in Dom(g)$ and $\alpha \in [0,1]$.

[2] It is easy to check that

$$\int_{[0,1]^n} M(\mathbf{x}) d\mathbf{x} = \frac{1}{n}\left(\int_0^1 x_1 dx_1 + \ldots + \int_0^1 x_n dx_n\right) = \frac{n}{n} \int_0^1 t dt = \frac{1}{2}.$$

Substituting the above value in (2.1) we obtain $orness(M) = \frac{1}{2}$. Following, for a weighted arithmetic mean we also obtain

$$\int_{[0,1]^n} M_{\mathbf{w}}(\mathbf{x}) d\mathbf{x} = w_1 \int_0^1 x_1 dx_1 + \ldots + w_n \int_0^1 x_n dx_n = \sum_{i=1}^{n} w_i \int_0^1 t dt = \frac{1}{2}.$$

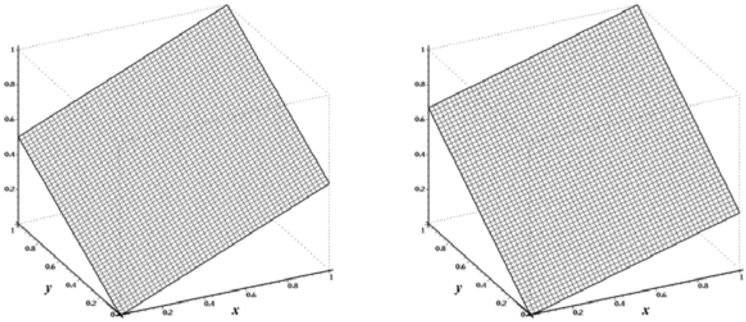


Fig. 2.1. 3D plots of weighted arithmetic means $M_{(\frac{1}{2},\frac{1}{2})}$ and $M_{(\frac{1}{3},\frac{2}{3})}$.

Definition 2.7 (Geometric mean). *The geometric mean is the function*

$$G(\mathbf{x}) = \sqrt[n]{x_1 x_2 \ldots x_n} = \left(\prod_{i=1}^{n} x_i\right)^{1/n}.$$

Definition 2.8 (Weighted geometric mean). *Given a weighting vector* $\mathbf{w}$, *the weighted geometric mean is the function*

$$G_{\mathbf{w}}(\mathbf{x}) = \prod_{i=1}^{n} x_i^{w_i}.$$

Definition 2.9 (Harmonic mean). *The harmonic mean is the function*

$$H(\mathbf{x}) = n\left(\sum_{i=1}^{n} \frac{1}{x_i}\right)^{-1}.$$

Definition 2.10 (Weighted harmonic mean). *Given a weighting vector* $\mathbf{w}$, *the weighted harmonic mean is the function*

$$H_{\mathbf{w}}(\mathbf{x}) = \left(\sum_{i=1}^{n} \frac{w_i}{x_i}\right)^{-1}.$$

Note 2.11. If the weighting vector $\mathbf{w}$ is given without normalization, i.e., $W = \sum_{i=1}^{n} w_i \neq 1$, then one can either normalize it first by dividing each component by W, or use the alternative expressions for weighted geometric and harmonic means

$$G_{\mathbf{w}}(\mathbf{x}) = \left(\prod_{i=1}^{n} x_i^{w_i} \right)^{1/W},$$

$$H_{\mathbf{w}}(\mathbf{x}) = W \left(\sum_{i=1}^{n} \frac{w_i}{x_i} \right)^{-1}.$$

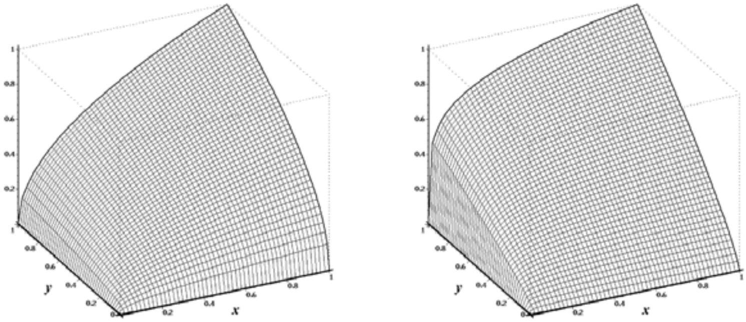


Fig. 2.2. 3D plots of weighted geometric means $G_{(\frac{1}{2},\frac{1}{2})}$ and $G_{(\frac{1}{5},\frac{4}{5})}$.

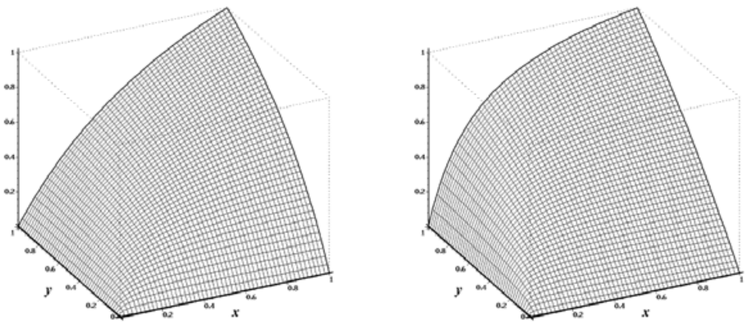


Fig. 2.3. 3D plots of weighted harmonic means $H_{(\frac{1}{2},\frac{1}{2})}$ and $H_{(\frac{1}{5},\frac{4}{5})}$.

Geometric-Arithmetic Mean Inequality

The following result is an extended version of the well known geometric-arithmetic means inequality

$$H_{\mathbf{w}}(\mathbf{x}) \le G_{\mathbf{w}}(\mathbf{x}) \le M_{\mathbf{w}}(\mathbf{x}), \tag{2.4}$$

for any vector $\mathbf{x}$ and weighting vector $\mathbf{w}$, with equality if and only if $\mathbf{x} = (t, t, \ldots, t)$.

Another curious relation between these three means is that for $n = 2$ we have $G(x, y) = \sqrt{M(x, y) \cdot H(x, y)}$.

Power means

A further generalization of the arithmetic mean is a family called power means (also called *root-power means*). This family is defined by

---

Definition 2.12 (Power mean). *For $r \in \Re$, the power mean is the function*

$$M_{[r]}(\mathbf{x}) = \left(\frac{1}{n} \sum_{i=1}^{n} x_i^r \right)^{1/r},$$

if $r \neq 0$, and $M_{[0]}(\mathbf{x}) = G(\mathbf{x})$.[3]

Definition 2.13 (Weighted power mean). *Given a weighting vector* $\mathbf{w}$ *and $r \in \Re$, the weighted power mean is the function*

$$M_{\mathbf{w},[r]}(\mathbf{x}) = \left(\sum_{i=1}^{n} w_i x_i^r \right)^{1/r},$$

if $r \neq 0$, and $M_{\mathbf{w},[0]}(\mathbf{x}) = G_{\mathbf{w}}(\mathbf{x})$.

Note 2.14. The family of weighted power means is *augmented* to $r = -\infty$ and $r = \infty$ by using the limiting cases

$$M_{\mathbf{w},[-\infty]}(\mathbf{x}) = \lim_{r \to -\infty} M_{\mathbf{w},[r]}(\mathbf{x}) = \min(\mathbf{x}),$$

$$M_{\mathbf{w},[\infty]}(\mathbf{x}) = \lim_{r \to \infty} M_{\mathbf{w},[r]}(\mathbf{x}) = \max(\mathbf{x}).$$

However min and max are not themselves power means.

The limiting case of the weighted geometric mean is also obtained as

[3] We shall use square brackets in the notation $M_{[r]}$ for power means to distinguish them from quasi-arithmetic means M_g (see Section 2.3), where parameter g denotes a generating function rather than a real number. The same applies to the weighted power means.

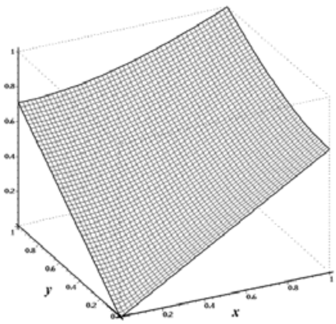
$$M_{\mathbf{w},[0]}(\mathbf{x}) = \lim_{r\to 0} M_{\mathbf{w},[r]}(\mathbf{x}) = G_{\mathbf{w}}(\mathbf{x}).$$

Of course, the family of weighted power means includes the special cases $M_{\mathbf{w},[1]}(\mathbf{x}) = M_{\mathbf{w}}(\mathbf{x})$, and $M_{\mathbf{w},[-1]}(\mathbf{x}) = H_{\mathbf{w}}(\mathbf{x})$. Another special case is the weighted *quadratic mean*

$$M_{\mathbf{w},[2]}(\mathbf{x}) = Q_{\mathbf{w}}(\mathbf{x}) = \sqrt{\sum_{i=1}^{n} w_i x_i^2}.$$

Main properties

- The weighted power mean $M_{\mathbf{w},[r]}$ is a strictly increasing aggregation function, if all $w_i > 0$ and $0 < r < \infty$;
- $M_{\mathbf{w},[r]}$ is a continuous function on $[0,1]^n$;
- $M_{\mathbf{w},[r]}$ is an asymmetric idempotent function (symmetric if all $w_i = \frac{1}{n}$);
- $M_{\mathbf{w},[r]}$ is a homogeneous function, i.e., $M_{\mathbf{w},[r]}(\lambda\mathbf{x}) = \lambda M_{\mathbf{w},[r]}(\mathbf{x})$ for all $\mathbf{x} \in [0,1]^n$ and for all $\lambda \in [0,1]$; it is the only homogeneous weighted quasi-arithmetic mean (this class is introduced in Section 2.3);
- Weighted power means are comparable: $M_{\mathbf{w},[r]}(\mathbf{x}) \le M_{\mathbf{w},[s]}(\mathbf{x})$ if $r \le s$; this implies the geometric-arithmetic mean inequality;
- $M_{\mathbf{w},[r]}$ has absorbing element (always $a = 0$) if and only if $r \le 0$ (and all weights w_i are positive);
- $M_{\mathbf{w},[r]}$ does not have a neutral element[4].



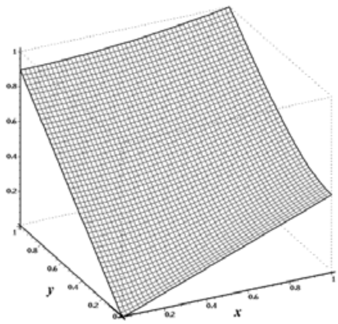


Fig. 2.4. 3D plots of weighted quadratic mean $Q_{(\frac{1}{2},\frac{1}{2})}$ and $Q_{(\frac{1}{5}\frac{4}{5})}$.

[4] The limiting cases min ($r = -\infty$) and max ($r = \infty$) which have neutral elements $e = 1$ and $e = 0$ respectively, are not themselves power means.

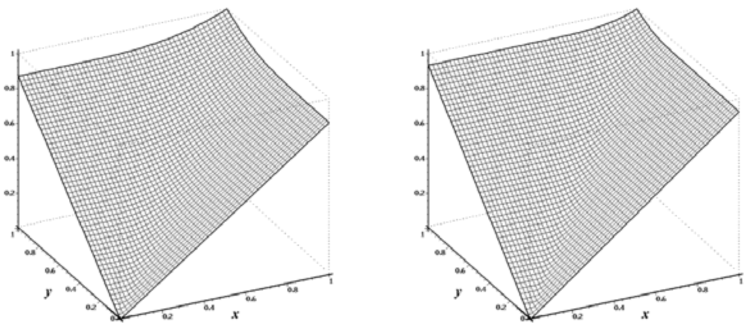


Fig. 2.5. 3D plots of power means $M_{[5]}$ and $M_{[10]}$.

Measure of orness

Calculations for the geometric mean yields

$$orness(G) = -\frac{1}{n-1} + \frac{n+1}{n-1}\left(\frac{n}{n+1}\right)^n,$$

but for other means explicit formulas are known only for special cases, e.g., $n = 2$

$$\int_{[0,1]^2} Q(\mathbf{x})d\mathbf{x} = \frac{1}{3}\left(1 + \frac{1}{\sqrt{2}}\ln(1+\sqrt{2})\right) \approx 0.541075,$$

$$\int_{[0,1]^2} H(\mathbf{x})d\mathbf{x} = \frac{4}{3}(1 - \ln(2)), \quad \text{and}$$

$$\int_{[0,1]^2} M_{[-2]}(\mathbf{x})d\mathbf{x} = \frac{2}{3}(2-\sqrt{2}),$$

from which, when $n = 2$, $orness(Q) \approx 0.623225$, $orness(H) \approx 0.22741$, and $orness(M_{[-2]}) = 3 - 2\sqrt{2}$.

Definition 2.15 (Dual weighted power mean). *Let $M_{\mathbf{w},[r]}$ be a weighted power mean. The function*

$$\bar{M}_{\mathbf{w},[r]}(\mathbf{x}) = 1 - M_{\mathbf{w},[r]}(\mathbf{1} - \mathbf{x})$$

is called the dual weighted power mean.

Note 2.16. The dual weighted power mean is obviously a mean (the class of means is closed under duality). The absorbent element, if any, becomes $a = 1$. The extensions of weighted power means satisfy $\bar{M}_{\mathbf{w},[\infty]}(\mathbf{x}) = M_{\mathbf{w},[-\infty]}(\mathbf{x})$ and $\bar{M}_{\mathbf{w},[-\infty]}(\mathbf{x}) = M_{\mathbf{w},[\infty]}(\mathbf{x})$. The weighted arithmetic mean $M_{\mathbf{w}}$ is self-dual.

2.3 Weighted quasi-arithmetic means

2.3.1 Definitions

Quasi-arithmetic means generalize power means. Consider a univariate continuous strictly monotone function $g : [0,1] \to [-\infty, \infty]$, which we call a *generating function.* Of course, g is invertible, but it is not necessarily a bijection (i.e., its range may be $Ran(g) \subset [-\infty, \infty]$).

Definition 2.17 (Quasi-arithmetic mean). *For a given generating function g, the quasi-arithmetic mean is the function*

$$M_g(\mathbf{x}) = g^{-1}\left(\frac{1}{n}\sum_{i=1}^{n} g(x_i)\right). \tag{2.5}$$

Its weighted analogue is given by

Definition 2.18 (Weighted quasi-arithmetic mean). *For a given generating function g, and a weighting vector $\mathbf{w}$, the weighted quasi-arithmetic mean is the function*

$$M_{\mathbf{w},g}(\mathbf{x}) = g^{-1}\left(\sum_{i=1}^{n} w_i g(x_i)\right). \tag{2.6}$$

The weighted power means are a subclass of weighted quasi-arithmetic means with the generating function

$$g(t) = \begin{cases} t^r, & \text{if } r \neq 0, \\ \log(t), & \text{if } r = 0. \end{cases}$$

Note 2.19. Observe that if $Ran(g) = [-\infty, \infty]$, then we have the summation $-\infty + \infty$ or $+\infty - \infty$ if $x_i = 0$ and $x_j = 1$ for some $i \neq j$. When this occurs, a convention, such as $-\infty + \infty = +\infty - \infty = -\infty$, is adopted, and continuity of $M_{\mathbf{w},g}$ is lost.

Note 2.20. If the weighting vector $\mathbf{w}$ is not normalized, i.e., $W = \sum_{i=1}^{n} w_i \neq 1$, then weighted quasi-arithmetic means are expressed as

$$M_{\mathbf{w},g}(\mathbf{x}) = g^{-1}\left(\frac{1}{W}\sum_{i=1}^{n} w_i g(x_i)\right).$$

2.3.2 Main properties

- Weighted quasi-arithmetic means are continuous if and only if $Ran(g) \neq [-\infty, \infty]$ [151];

- Weighted quasi-arithmetic means with strictly positive weights are strictly monotone increasing on $]0, 1[^n$;
- The class of weighted quasi-arithmetic means is closed under duality. That is, given a strong negation N, the N-dual of a weighted quasi-arithmetic mean $M_{\mathbf{w},g}$ is in turn a weighted quasi-arithmetic mean, given by $M_{\mathbf{w},g\circ N}$. For the standard negation, the dual of a weighted quasi-arithmetic mean is characterized by the generating function $h(t) = g(1-t)$;
- The following result regarding self-duality holds (see, e.g., [209]): Given a strong negation N, a weighted quasi-arithmetic mean $M_{\mathbf{w},g}$ is N-self-dual if and only if N is the strong negation generated by g, i.e., if $N(t) = g^{-1}(g(0) + g(1) - g(t))$ for any $t \in [0, 1]$. This implies, in particular:
  - Weighted quasi-arithmetic means, such that $g(0) = \pm\infty$ or $g(1) = \pm\infty$ are never N-self-dual (in fact, they are dual to each other);
  - Weighted arithmetic means are always self-dual (i.e., N-self-dual with respect to the standard negation $N(t) = 1 - t$);
- The generating function is not defined uniquely, but up to an arbitrary linear transformation, i.e., if $g(t)$ is a generating function of some weighted quasi-arithmetic mean, then $ag(t) + b$, $a, b \in \Re$, $a \neq 0$ is also a generating function of *the same mean*[5], provided $Ran(g) \neq [-\infty, \infty]$;
- There are incomparable quasi-arithmetic means. Two quasi-arithmetic means M_g and M_h satisfy $M_g \leq M_h$ if and only if either the composite $g \circ h^{-1}$ is convex and g is decreasing, or $g \circ h^{-1}$ is concave and g increasing;
- The only homogeneous weighted quasi-arithmetic means are weighted power means;
- Weighted quasi-arithmetic means do not have a neutral element[6]. They may have an absorbing element only when all the weights are strictly positive and $g(0) = \pm\infty$ or $g(1) = \pm\infty$, and in such cases the corresponding absorbing elements are, respectively, $a = 0$ and $a = 1$.

2.3.3 Examples

Example 2.21 (Weighted power means). Weighted power means are a special case of weighted quasi-arithmetic means, with $g(t) = t^r$, $r \neq 0$ and $g(t) = \log(t)$ if $r = 0$. Note that the generating function $g(t) = \frac{t^r - 1}{r}$ defines exactly the same power mean (as a particular case of a linear transformation of g). Also note the similarity of the latter to the additive generators of the Schweizer-Sklar family of triangular norms, p. 150. By taking $g(t) = (1-t)^r$, $r \neq 0$ and $g(t) = \log(1-t)$ if $r = 0$, we obtain the family of dual weighted power means, which are related to Yager triangular norms, p. 156.

---

[5] For this reason, one can assume that g is monotone increasing, as otherwise we can simply take $-g$.

[6] Observe that the limiting cases min and max are not quasi-arithmetic means.

Example 2.22 (Harmonic and geometric means). These classical means, defined on page 43, are special cases of the power means, obtained when $g(t) = t^r$, $r = -1$ and $r = 0$ respectively.

Example 2.23. Let $g(t) = \log \frac{t}{1-t}$. The corresponding quasi–arithmetic mean M_g is given by

$$M_g(\mathbf{x}) = \begin{cases} \dfrac{\sqrt[n]{\prod_{i=1}^{n} x_i}}{\sqrt[n]{\prod_{i=1}^{n} x_i} + \sqrt[n]{\prod_{i=1}^{n} (1-x_i)}}, & \text{if } \{0,1\} \nsubseteq \{x_1, \ldots, x_n\} \\ 0, & \text{otherwise,} \end{cases}$$

that is, $M_g = \frac{G}{G - G_d + 1}$ and with the convention $\frac{0}{0} = 0$.

Example 2.24 (Weighted trigonometric means). Let $g_1(t) = \sin(\frac{\pi}{2}t)$, $g_2(t) = \cos(\frac{\pi}{2}t)$, and $g_3(t) = \tan(\frac{\pi}{2}t)$ be the generating functions. The weighted trigonometric means are the functions

$$SM_{\mathbf{w}}(\mathbf{x}) = \frac{2}{\pi} \arcsin(\sum_{i=1}^{n} w_i \sin(\frac{\pi}{2} x_i)),$$

$$CM_{\mathbf{w}}(\mathbf{x}) = \frac{2}{\pi} \arccos(\sum_{i=1}^{n} w_i \cos(\frac{\pi}{2} x_i)) \text{ and}$$

$$TM_{\mathbf{w}}(\mathbf{x}) = \frac{2}{\pi} \arctan(\sum_{i=1}^{n} w_i \tan(\frac{\pi}{2} x_i)).$$

Their 3D plots are presented on Figures 2.6 and 2.7.

Example 2.25 (Weighted exponential means). Let the generating function be

$$g(t) = \begin{cases} \gamma^t, & \text{if } \gamma \neq 1, \\ t, & \text{if } \gamma = 1. \end{cases}$$

The weighted exponential mean is the function

$$EM_{\mathbf{w},\gamma}(\mathbf{x}) = \begin{cases} \log_\gamma \left(\sum_{i=1}^{n} w_i \gamma^{x_i} \right), & \text{if } \gamma \neq 1, \\ M_{\mathbf{w}}(\mathbf{x}), & \text{if } \gamma = 1. \end{cases}$$

3D plots of some weighted exponential means are presented on Figures 2.8 and 2.9.

Example 2.26. There is another mean also known as exponential [40], given for $\mathbf{x} \geq \mathbf{1}$ by

$$f(\mathbf{x}) = \exp\left((\prod_{i=1}^{n} \log(x_i))^{1/n} \right).$$

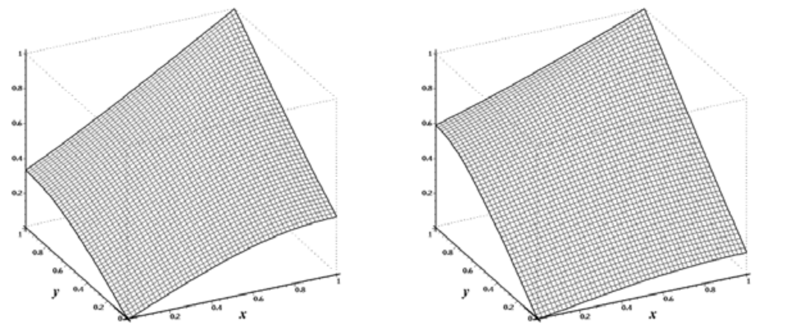


Fig. 2.6. 3D plots of weighted trigonometric means $SM_{(\frac{1}{2},\frac{1}{2})}$ and $SM_{(\frac{1}{5},\frac{4}{5})}$.

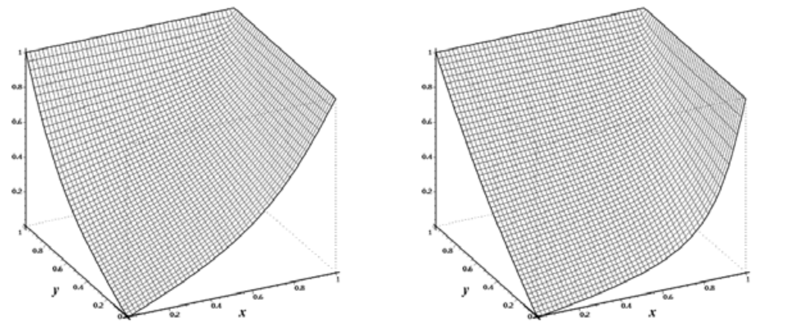


Fig. 2.7. 3D plots of weighted trigonometric means $TM_{(\frac{1}{2},\frac{1}{2})}$ and $TM_{(\frac{1}{5},\frac{4}{5})}$.

It is a quasi-arithmetic mean with a generating function $g(t) = \log(\log(t))$, and its inverse $g^{-1}(t) = \exp(\exp(t))$.

In the domain $[0,1]^n$ one can use a generating function $g(t) = \log(-\log(t))$, so that its inverse is $g^{-1}(t) = \exp(-\exp(t))$. This mean is discontinuous, since $Ran(g) = [-\infty, \infty]$. We obtain the expression

$$f(\mathbf{x}) = \exp\left(-\prod_{i=1}^{n}(-\log(x_i))^{1/n}\right).$$

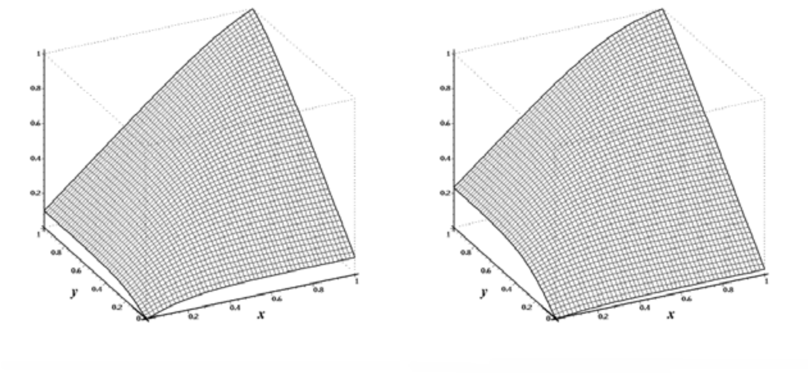


Fig. 2.8. 3D plots of weighted exponential means $EM_{(\frac{1}{2},\frac{1}{2}),0.001}$ and $EM_{(\frac{1}{5},\frac{4}{5}),0.001}$.

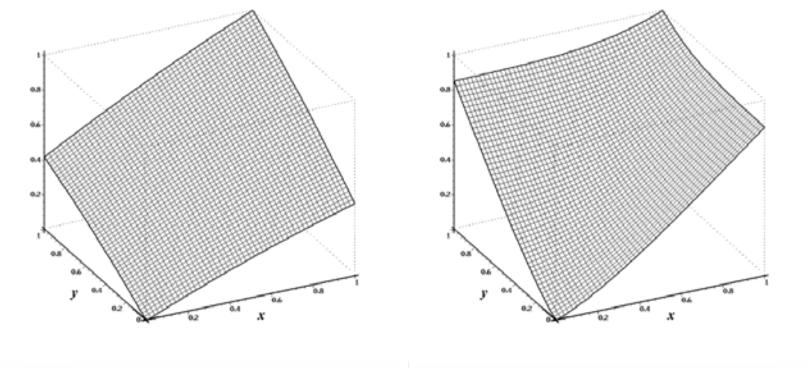


Fig. 2.9. 3D plots of exponential means $EM_{(\frac{1}{2},\frac{1}{2}),0.5}$ and $EM_{(\frac{1}{2},\frac{1}{2}),100}$.

Example 2.27 (Weighted radical means). Let $\gamma > 0$, $\gamma \neq 1$, and let the generating function be

$$g(t) = \gamma^{1/t}.$$

The weighted radical mean is the function

$$RM_{\mathbf{w},\gamma}(\mathbf{x}) = \left(\log_\gamma (\sum_{i=1}^n w_i \gamma^{1/x_i}) \right)^{-1}.$$

3D plots of some radical means are presented on Figure 2.10.

Example 2.28 (Weighted basis-exponential means). Weighted basis-exponential means are obtained by using the generating function $g(t) = t^t$ and $t \geq \frac{1}{e}$ (this

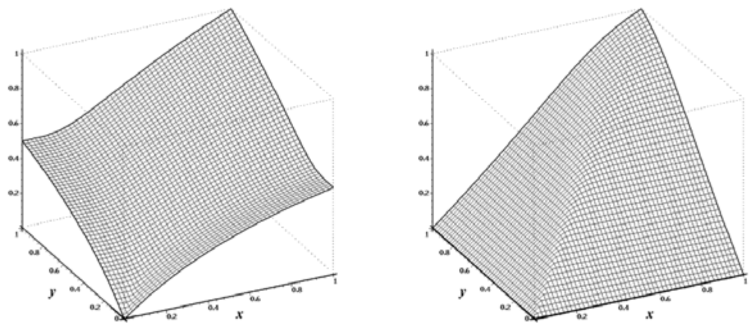


Fig. 2.10. 3D plots of radical means $RM_{(\frac{1}{2},\frac{1}{2}),0.5}$ and $RM_{(\frac{1}{2},\frac{1}{2}),100}$.

generating function is decreasing on $[0, \frac{1}{e}[$ and increasing on $]\frac{1}{e}, \infty]$, hence the restriction). The value of this mean is such a value y that

$$y^y = \sum_{i=1}^{n} w_i x_i^{x_i}.$$

For practical purposes this equation has to be solved for y numerically.

Example 2.29 (Weighted basis-radical means). Weighted basis-radical means are obtained by using the generator $g(t) = t^{1/t}$ and $t \geq \frac{1}{e}$ (restriction for the same reason as in the Example 2.28). The value of this mean is such a value y that

$$y^{1/y} = \sum_{i=1}^{n} w_i x_i^{1/x_i}.$$

For practical purposes this equation has to be solved for y numerically.

2.3.4 Calculation

Generating functions offer a nice way of calculating the values of weighted quasi-arithmetic means. Note that we can write

$$M_{\mathbf{w},g}(\mathbf{x}) = g^{-1}\left(M_{\mathbf{w}}(g(\mathbf{x}))\right),$$

where $g(\mathbf{x}) = (g(x_1), \ldots, g(x_n))$. Thus calculation can be performed in three steps:

1. Transform all the inputs by calculating vector $g(\mathbf{x})$;
2. Calculate the weighted arithmetic mean of the transformed inputs;
3. Calculate the inverse g^{-1} of the computed mean.

However one needs to be careful with the limiting cases, for example when $g(x_i)$ becomes infinite. Typically this is an indication of existence of an absorbing element, this needs to be picked up by the computer subroutine. Similarly, special cases like $M_{[r]}(\mathbf{x})$, $r \to \pm\infty$ have to be accommodated (in these cases the subroutine has to return the minimum or the maximum).

2.3.5 Weighting triangles

When we are interested in using weighted quasi-arithmetic means as extended aggregation functions, we need to have a clear rule as to how the weighting vectors are calculated for each dimension $n = 2, 3, \ldots$. For symmetric quasi-arithmetic means we have a simple rule: for each n the weighting vector $\mathbf{w}^n = (\frac{1}{n}, \ldots, \frac{1}{n})$. For weighted means we need the concept of a weighting triangle.

Definition 2.30 (Weighting triangle). *A weighting triangle or triangle of weights is a set of numbers $w_i^n \in [0, 1]$, for $i = 1, \ldots, n$ and $n \geq 1$, such that: $\sum\limits_{i=1}^{n} w_i^n = 1$, for all $n \geq 1$. It will be represented in the following form*

$$\begin{array}{ccccccc} & & & 1 & & & \\ & & w_1^2 & & w_2^2 & & \\ & w_1^3 & & w_2^3 & & w_3^3 & \\ w_1^4 & & w_2^4 & & w_3^4 & & w_4^4 \\ & & & \ldots & & & \end{array}$$

Weighting triangles will be denoted by $\triangle w_i^n$.

Example 2.31. A basic example is the "normalized" Pascal triangle

$$\begin{array}{ccccccccc} & & & & 1 & & & & \\ & & & 1/2 & & 1/2 & & & \\ & & 1/4 & & 2/4 & & 1/4 & & \\ & 1/8 & & 3/8 & & 3/8 & & 1/8 & \\ 1/16 & & 4/16 & & 6/16 & & 4/16 & & 1/16 \\ & & & & \ldots\ldots & & & & \end{array}$$

The generic formula for the weighting vector of dimension n in this weighting triangle is

$$\mathbf{w}^n = \frac{1}{2^{n-1}} \left(\binom{n-1}{0}, \binom{n-1}{1}, \ldots, \binom{n-1}{n-1} \right)$$

for each $n \geq 1$.[7]

It is possible to generate weighting triangles in different ways [44]:

[7] Recall $\binom{n}{m} = \frac{n!}{m!(n-m)!}$.

Proposition 2.32. *The following methods generate weighting triangles:*

1. *Let $\lambda_1, \lambda_2, \ldots \geq 0$ be a sequence of non-negative real numbers such that $\lambda_1 > 0$. Define the weights using*
$$w_i^n = \frac{\lambda_{n-i+1}}{\lambda_1 + \cdots + \lambda_n},$$
for all $i = 1, \ldots, n$ and $n \geq 1$;
2. *Let N be a strong negation.* [8] *Generate the weights using N by*
$$w_i^n = N\left(\frac{i-1}{n}\right) - N\left(\frac{i}{n}\right),$$
for all $i = 1, \ldots, n$ and $n \geq 1$;
3. *Let Q be a monotone non-decreasing function $Q : [0,1] \to [0,1]$ such that*[9] *$Q(0) = 0$ and $Q(1) = 1$. Generate the weights using function Q by [264]*
$$w_i^n = Q\left(\frac{i}{n}\right) - Q\left(\frac{i-1}{n}\right),$$
for all $i = 1, \ldots, n$ and $n \geq 1$.

Another way to construct weighting triangles is by using fractal structures exemplified below. Such weighting triangles cannot be generated by any of the methods in Proposition 2.32.

Example 2.33. The following two triangles belong to the Sierpinski family [44]

$$\begin{array}{c}
1 \\
1.\frac{1}{4} \quad 3.\frac{1}{4} \\
1.\frac{1}{4} \quad 3.\frac{1}{4^2} \quad 3^2.\frac{1}{4^2} \\
1.\frac{1}{4} \quad 3.\frac{1}{4^2} \quad 3^2.\frac{1}{4^3} \quad 3^3.\frac{1}{4^3} \\
\cdots\cdots\cdots
\end{array}$$

and

$$\begin{array}{c}
1 \\
1 \cdot \frac{1}{9} \quad 8 \cdot \frac{1}{9} \\
1 \cdot \frac{1}{9} \quad 8 \cdot \frac{1}{9^2} \quad 8^2 \cdot \frac{1}{9^2} \\
1 \cdot \frac{1}{9} \quad 8 \cdot \frac{1}{9^2} \quad 8^2 \cdot \frac{1}{9^3} \quad 8^3 \cdot \frac{1}{9^3} \\
\cdots\cdots\cdots
\end{array}$$

[8] See Definition 1.48 on p. 18.

[9] This is a so-called quantifier, see p. 81.

In general, given two real numbers a, b such that $a < b$ and $\frac{a}{b} \neq \frac{b-a}{b}$, it is possible to define a weighting triangle by [44]

$$\begin{array}{ccccccc} & & & 1 & & & \\ & & \frac{a}{b} & & \frac{b-a}{b} & & \\ & \frac{a}{b} & & (\frac{b-a}{b})(\frac{a}{b}) & & (\frac{b-a}{b})^2 & \\ \frac{a}{b} & & (\frac{b-a}{b})(\frac{a}{b}) & & (\frac{b-a}{b})^2(\frac{a}{b}) & & (\frac{b-a}{b})^3 \\ & & & \dots & & & \end{array}$$

This weighting triangle also belongs to the Sierpinski family, and a generic formula for the weights is

$$w_i^n = \frac{a}{b} \cdot \left(\frac{b-a}{b}\right)^{i-1}, \quad i = 1, \dots, n-1 \quad \text{and} \quad w_n^n = \left(\frac{b-a}{b}\right)^{n-1}.$$

Let us now mention two characterization theorems, which relate continuity, strict monotonicity and the properties of decomposability and bisymmetry to the class of weighted quasi-arithmetic means.

Theorem 2.34 (Kolmogorov-Nagumo). *An extended aggregation function F is continuous, decomposable* [10]*, and strictly monotone if and only if there is a monotone bijection $g : [0, 1] \to [0, 1]$, such that for each $n > 1$, f_n is a quasi-arithmetic mean M_g.*

The next result is a generalized version of Kolmogorov and Nagumo characterization, due to Aczél [1]

Theorem 2.35. *An extended aggregation function F is continuous, bisymmetric* [11]*, idempotent, and strictly monotone if and only if there is a monotone bijection $g : [0, 1] \to [0, 1]$, and a weighting triangle $\triangle w_i^n$ with all positive weights, so that for each $n > 1$, f_n is a weighted quasi-arithmetic mean $M_{\mathbf{w}^n,g}$ (i.e., $f_n = M_{\mathbf{w}^n,g}$).*

Note 2.36. If we omit the strict monotonicity of F, we recover the class of non–strict means introduced by Fodor and Marichal [96].

2.3.6 Weights dispersion

An important quantity associated with weighting vectors is their dispersion, also called entropy.

[10] See Definition 1.42. Continuity and decomposability imply idempotency.
[11] See Definition 1.43.

Definition 2.37 (Weights dispersion (entropy)). *For a given weighting vector* $\mathbf{w}$ *its measure of dispersion (entropy) is*

$$Disp(\mathbf{w}) = -\sum_{i=1}^{n} w_i \log w_i, \tag{2.7}$$

with the convention $0 \cdot \log 0 = 0$.

The weights dispersion measures the degree to which a weighted aggregation function f takes into account all inputs. For example, in the case of weighted means, among the two weighting vectors $\mathbf{w}_1 = (0, 1)$ and $\mathbf{w}_2 = (0.5, 0.5)$ the second one may be preferable, since the corresponding weighted mean uses information from two sources rather than a single source, and is consequently less sensitive to input inaccuracies.

A useful normalization of this measure is

$$-\frac{1}{\log n} \sum_{i=1}^{n} w_i \log w_i.$$

Along with the orness value (p. 40), the weights entropy is an important parameter in choosing weighting vectors of both quasi-arithmetic means and OWA functions (see p. 70).

There are also other entropy measures (e.g., Rényi entropy) frequently used in studies of weighted aggregation functions, e.g., [266] [12].

2.3.7 How to choose weights

Choosing weights of weighted arithmetic means

In each application the weighting vector of the weighted arithmetic mean will be different. We examine the problem of choosing the weighting vector which fits best some empirical data, the pairs $(\mathbf{x}_k, y_k), k = 1, \ldots, K$. Our goal is to determine the best weighted arithmetic mean that minimizes the norm of the differences between the predicted $(f(\mathbf{x}_k))$ and observed (y_k) values. We will use the least squares or least absolute deviation criterion, as discussed on p. 33. In the first case we have the following optimization problem

$$\begin{aligned} \min \quad & \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i x_{ik} - y_k \right)^2 \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n. \end{aligned} \tag{2.8}$$

[12] These measures of entropy can be obtained by relaxing the subadditivity condition which characterizes Shannon entropy [241].

It is easy to recognize a standard quadratic programming problem (QP), with a convex objective function. There are plenty of standard methods for its solution, discussed in the Appendix A.5.

We mentioned on p. 33 that one can use a different fitting criterion, such as the least absolute deviation (LAD) criterion, which translates into a different optimization problem

$$\begin{aligned} \min \quad & \sum_{k=1}^{K} \left| \sum_{i=1}^{n} w_i x_{ik} - y_k \right| \qquad (2.9) \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n. \end{aligned}$$

This problem is subsequently converted into a linear programming problem (LP) as discussed in the Appendix A.2.

Particular attention is needed for the case when the quadratic (resp. linear) programming problems have singular matrices. Such cases appear when there are few data, or when the input values are linearly dependent. While modern quadratic and linear programming methods accommodate for such cases, the minimization problem will typically have multiple solutions. An additional criterion is then used to select one of these solutions, and typically this criterion relates to the dispersion of weights, or the entropy [235], as defined in Definition 2.37. Torra [235] proposes to solve an auxiliary univariate optimization problem to maximize weights dispersion, subject to a given value of (2.8).

Specifically, one solves the problem

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} w_i \log w_i \qquad (2.10) \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n, \\ & \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i x_{ik} - y_k \right)^2 = A, \end{aligned}$$

where A is the value of the solution of problem (2.8). It turns out that if Problem (2.8) has multiple solutions, they are expressed in parametric form as linear combinations of one another. Further, the objective function in (2.10) is convex. Therefore problem (2.10) is a convex programming problem subject to linear constraints, and it can be solved by standard methods, see [235].

A different additional criterion is the so-called measure of orness (discussed in Section 2.1), which measures how far a given averaging function is from the max function, which is the weakest disjunctive function. It is applicable to any averaging function, and is frequently used as an additional constraint or criterion when constructing these functions. However, for any weighted arithmetic mean, the measure of orness is always $\frac{1}{2}$, therefore this parameter does not discriminate between arithmetic means with different weighting vectors.

Preservation of ordering of the outputs

We recall from Section 1.6, p. 34, that sometimes one not only has to fit an aggregation function to the numerical data, but also preserve the ordering of the outputs. That is, if $y_j \leq y_k$ then we expect $f(\mathbf{x}_j) \leq f(\mathbf{x}_k)$.

First, arrange the data, so that the outputs are in non-decreasing order, i.e., $y_k \leq y_{k+1}, k = 1, \ldots, K-1$. Define the additional linear constraints

$$< \mathbf{x}_{k+1} - \mathbf{x}_k, \mathbf{w} >= \sum_{i=1}^{n} w_i (x_{i,k+1} - x_{ik}) \geq 0,$$

$k = 1, \ldots, K-1$. We add the above constraints to problem (2.8) or (2.9) and solve it. The addition of an extra $K-1$ constraints neither changes the structure of the optimization problem, nor drastically affects its complexity.

Choosing weights of weighted quasi-arithmetic means

Consider the case of weighted quasi-arithmetic means, when a given generating function g is given. As before, we have a data set $(\mathbf{x}_k, y_k), k = 1, \ldots, K$, and we are interested in finding the weighting vector $\mathbf{w}$ that fits the data best. When we use the least squares, as discussed on p. 33, we have the following optimization problem

$$\begin{aligned} \min \ & \sum_{k=1}^{K} \left(g^{-1} \left(\sum_{i=1}^{n} w_i g(x_{ik}) \right) - y_k \right)^2 \\ \text{s.t.} \ & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n. \end{aligned} \tag{2.11}$$

This is a nonlinear optimization problem, but it can be reduced to quadratic programming by the following artifice. Let us apply g to y_k and the inner sum in (2.11). We obtain

$$\begin{aligned} \min \ & \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i g(x_{ik}) - g(y_k) \right)^2 \\ \text{s.t.} \ & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n. \end{aligned} \tag{2.12}$$

We recognize a standard quadratic programming problem (QP), with a convex objective function. This approach was discussed in detail in [17, 20, 30, 235]. There are plenty of standard methods of solution, discussed in the Appendix A.5.

If one uses the least absolute deviation (LAD) criterion (p. 33) we obtain a different optimization problem

$$\begin{aligned} \min \quad & \sum_{k=1}^{K} \left| \sum_{i=1}^{n} w_i g(x_{ik}) - g(y_k) \right| \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n. \end{aligned} \tag{2.13}$$

This problem is subsequently converted into a linear programming problem (LP) as discussed in the Appendix A.2.

As in the case of weighted arithmetic means, in the presence of multiple optimal solutions, one can use an additional criterion of the dispersion of weights [235].

Preservation of ordering of the outputs

Similarly to what we did for weighted arithmetic means (see also Section 1.6, p. 34), we will require that the ordering of the outputs is preserved, i.e., if $y_j \leq y_k$ then we expect $f(\mathbf{x}_j) \leq f(\mathbf{x}_k)$. We arrange the data, so that the outputs are in non-decreasing order, $y_k \leq y_{k+1}, k = 1, \ldots, K-1$. Then we define the additional linear constraints

$$< \mathbf{g}(x_{k+1}) - g(\mathbf{x}_k), \mathbf{w} >= \sum_{i=1}^{n} w_i (g(x_{i,k+1}) - g(x_{ik})) \geq 0,$$

$k = 1, \ldots, K-1$. We add the above constraints to problem (2.12) or (2.13) and solve it. The addition of extra $K-1$ constraints does not change the structure of the optimization problem, nor drastically affects its complexity.

Choosing generating functions

Consider now the case when the generating function g is also unknown, and hence needs to be found based on the data. We study two cases: a) when g is given algebraically, with one or more unknown parameters to estimate (e.g., $g_p(t) = t^p$, p unknown) and b) when no specific algebraic form of g is given.

In the first case we solve the problem

$$\begin{aligned} \min_{p,\mathbf{w}} \; & \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i g_p(x_{ik}) - g_p(y_k) \right)^2 \\ \text{s.t.} \; & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n, \\ & \text{conditions on } p. \end{aligned} \tag{2.14}$$

While this general optimization problem is non-convex and nonlinear (i.e., difficult to solve), we can convert it to a bi-level optimization problem (see Appendix A.5.3)

$$\min_{p} \left[\min_{\mathbf{w}} \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i g_p(x_{ik}) - g_p(y_k) \right)^2 \right] \tag{2.15}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n,$$

$$\text{plus conditions on } p.$$

The problem at the inner level is the same as (2.12) with a fixed g_p, which is a QP problem. At the outer level we have a global optimization problem with respect to a single parameter p. It is solved by using one of the methods discussed in Appendix A.5.4-A.5.5. We recommend deterministic Pijavski-Shubert method.

Example 2.38. Determine the weights and the generating function of a family of weighted power means. We have $g_p(t) = t^p$, and hence solve bi-level optimization problem

$$\min_{p \in [-\infty, \infty]} \left[\min_{\mathbf{w}} \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i x_{ik}^p - y_k^p \right)^2 \right] \tag{2.16}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n.$$

Of course, for numerical purposes we need to limit the range for p to a finite interval, and treat all the limiting cases $p \to \pm\infty$, $p \to 0$ and $p \to -1$.

A different situation arises when the parametric form of g is not given. The approach proposed in [17] is based on approximation of g with a monotone linear spline (see Appendix A.3, p. 309), as

$$g(t) = \sum_{j=1}^{J} c_j B_j(t), \tag{2.17}$$

where B_j are appropriately chosen basis functions, and c_j are spline coefficients. The monotonicity of g is ensured by imposing linear restrictions on spline coefficients, in particular non-negativity, as in [13]. Further, since the generating function is defined up to an arbitrary linear transformation, one has to fix a particular g by specifying two interpolation conditions, like $g(a) = 0, g(b) = 1, a, b \in]0, 1[$, and if necessary, properly model asymptotic behavior if $g(0)$ or $g(1)$ are infinite.

After rearranging the terms of the sum, the problem of identification becomes (subject to linear conditions on $\mathbf{c}, \mathbf{w}$)

$$\min_{\mathbf{c}, \mathbf{w}} \sum_{k=1}^{K} \left(\sum_{j=1}^{J} c_j \left[\sum_{i=1}^{n} w_i B_j(x_{ik}) - B_j(y_k) \right] \right)^2 . \tag{2.18}$$

For a fixed $\mathbf{c}$ (i.e., fixed g) we have a quadratic programming problem to find $\mathbf{w}$, and for a fixed $\mathbf{w}$, we have a quadratic programming problem to find $\mathbf{c}$. However if we consider both $\mathbf{c}, \mathbf{w}$ as variables, we obtain a difficult global optimization problem. We convert it into a bi-level optimization problem

$$\min_{\mathbf{c}} \min_{\mathbf{w}} \sum_{k=1}^{K} \left(\sum_{j=1}^{J} c_j \left[\sum_{i=1}^{n} w_i B_j(x_{ik}) - B_j(y_k) \right] \right)^2, \tag{2.19}$$

where at the inner level we have a QP problem and at the outer level we have a nonlinear problem with multiple local minima. When the number of spline coefficients J is not very large (< 10), this problem can be efficiently solved by using deterministic global optimization methods from Appendix A.5.5. If the number of variables is small and J is large, then reversing the order of minimization (i.e., using $\min_{\mathbf{w}} \min_{\mathbf{c}}$) is more efficient.

2.4 Other means

Besides weighted quasi-arithmetic means, there exist very large families of other means, some of which we will mention in this section. A comprehensive reference to the topic of means is [40]. However we must note that not all these means are monotone functions, so they cannot be used as aggregation functions. Still some members of these families are aggregation functions, and we will mention the sufficient conditions for monotonicity, if available. Most of the mentioned means do not require $\mathbf{x} \in [0,1]^n$, but we will assume $\mathbf{x} \geq \mathbf{0}$.

2.4.1 Gini means

Definition 2.39 (Gini mean). *Let $p, q \in \Re$ and $\mathbf{w} \in \Re^n, \mathbf{w} \geq 0$. Weighted Gini mean is the function*

$$G_{\mathbf{w}}^{p,q}(\mathbf{x}) = \begin{cases} \left(\dfrac{\sum\limits_{i=1}^{n} w_i x_i^p}{\sum\limits_{i=1}^{n} w_i x_i^q} \right)^{1/p-q} & \text{if } p \neq q, \\ \left(\prod\limits_{i=1}^{n} x_i^{w_i x_i^p} \right)^{1/\sum\limits_{i=1}^{n} w_i x_i^p} & \text{if } p = q. \end{cases} \tag{2.20}$$

Properties

- $G_{\mathbf{w}}^{p,q} = G_{\mathbf{w}}^{q,p}$, so we assume $p \geq q$;
- $\lim\limits_{p \to q} G_{\mathbf{w}}^{p,q} = G_{\mathbf{w}}^{q,q}$;
- $\lim\limits_{p \to \infty} G_{\mathbf{w}}^{p,q}(\mathbf{x}) = \max(\mathbf{x})$;

- $\lim_{q \to -\infty} G_{\mathbf{w}}^{p,q}(\mathbf{x}) = \min(\mathbf{x})$;
- If $p_1 \le p_2, q_1 \le q_2$, then $G_{\mathbf{w}}^{p_1,q_1} \le G_{\mathbf{w}}^{p_2,q_2}$.

Special cases

- Setting $q = 0$ and $p \ge 0$ leads to weighted power means $G_{\mathbf{w}}^{p,0} = M_{\mathbf{w},[p]}$.
- Setting $p = 0$ and $q \le 0$ also leads to weighted power means $G_{\mathbf{w}}^{0,q} = M_{\mathbf{w},[q]}$.
- Setting $q = p - 1$ leads to *counter-harmonic* means, also called *Lehmer means*. For example, when $n = 2$, $G_{(\frac{1}{2},\frac{1}{2})}^{q+1,q}(x_1, x_2) = \frac{x_1^{q+1}+x_2^{q+1}}{x_1^q+x_2^q}$, $q \in \Re$.
- When $q = 1$ we obtain the *contraharmonic* mean $G_{(\frac{1}{2},\frac{1}{2})}^{2,1}(x_1, x_2) = \frac{x_1^2+x_2^2}{x_1+x_2}$.

Note 2.40. Counter-harmonic means (and hence Gini means in general) are not monotone, except special cases (e.g., power means).

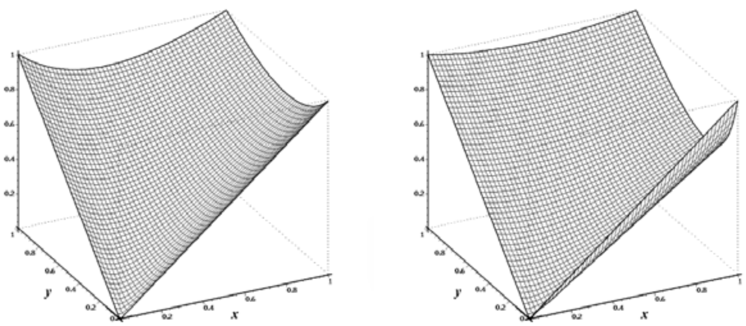


Fig. 2.11. 3D plots of weighted Gini means $G_{(\frac{1}{2},\frac{1}{2})}^{2,1}$ and $G_{(\frac{1}{5},\frac{4}{5})}^{2,1}$ (both are weighted contraharmonic means). Note lack of monotonicity.

2.4.2 Bonferroni means

Definition 2.41 (Bonferroni mean [36]). *Let* $p, q \ge 0$ *and* $\mathbf{x} \ge \mathbf{0}$. *Bonferroni mean is the function*

$$B^{p,q}(\mathbf{x}) = \left(\frac{1}{n(n-1)} \sum_{i,j=1, i \neq j}^{n} x_i^p x_j^q \right)^{1/(p+q)}. \tag{2.21}$$

Extension to $B^{p,q,r}(\mathbf{x})$*, etc., is obvious.*

It is an aggregation function.

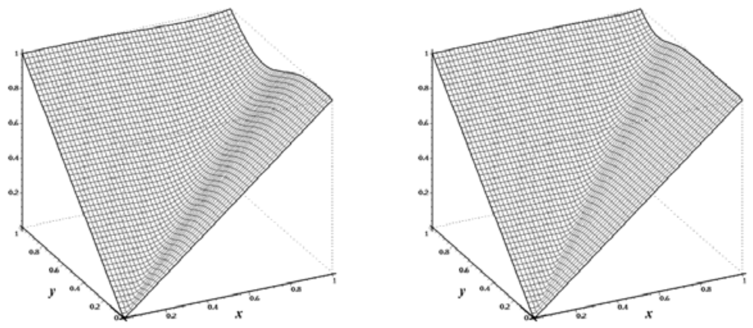


Fig. 2.12. 3D plots of weighted Gini means $G^{5,4}_{(\frac{1}{5},\frac{4}{5})}$ and $G^{10,9}_{(\frac{1}{5},\frac{4}{5})}$ (both are weighted counter-harmonic means).

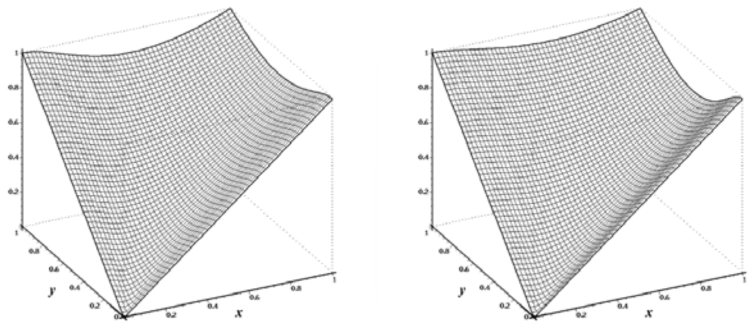


Fig. 2.13. 3D plot of weighted Gini means $G^{2,2}_{(\frac{1}{2},\frac{1}{2})}$ and $G^{2,2}_{(\frac{1}{3},\frac{2}{3})}$.

2.4.3 Heronian mean

Definition 2.42 (Heronian mean). *Heronian mean is the function*

$$HR(\mathbf{x}) = \frac{2}{n(n+1)} \sum_{i=1}^{n} \sum_{j=i}^{n} \sqrt{x_i x_j}. \tag{2.22}$$

It is an aggregation function. For $n = 2$ we have $HR = \frac{1}{3}(2M + G)$.

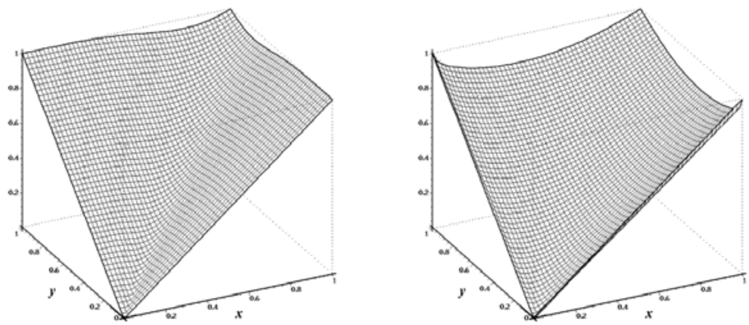


Fig. 2.14. 3D plots of weighted Gini means $G^{5,5}_{(\frac{1}{2},\frac{1}{2})}$ and $G^{3,0.5}_{(\frac{1}{2},\frac{1}{2})}$.

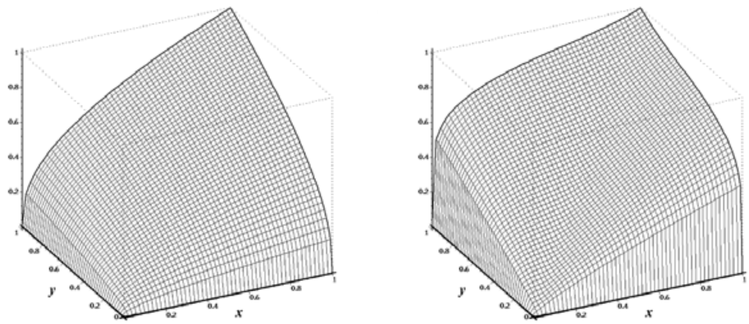


Fig. 2.15. 3D plots of Bonferroni means $B^{3,2}$ and $B^{10,2}$.

2.4.4 Generalized logarithmic means

Definition 2.43 (Generalized logarithmic mean). *Let $n = 2$, $x, y > 0$, $x \neq y$ and $p \in [-\infty, \infty]$. The generalized logarithmic mean is the function*

$$L^p(x,y) = \begin{cases} \frac{y-x}{\log y - \log x}, & \text{if } p = -1, \\ \frac{1}{e}\left(\frac{y^y}{x^x}\right)^{1/(y-x)}, & \text{if } p = 0, \\ \min(x,y), & \text{if } p = -\infty, \\ \max(x,y), & \text{if } p = \infty, \\ \left(\frac{y^{p+1}-x^{p+1}}{(p+1)(y-x)}\right)^{1/p} & \text{otherwise.} \end{cases} \tag{2.23}$$

For $x = y$, $L^p(x,x) = x$.

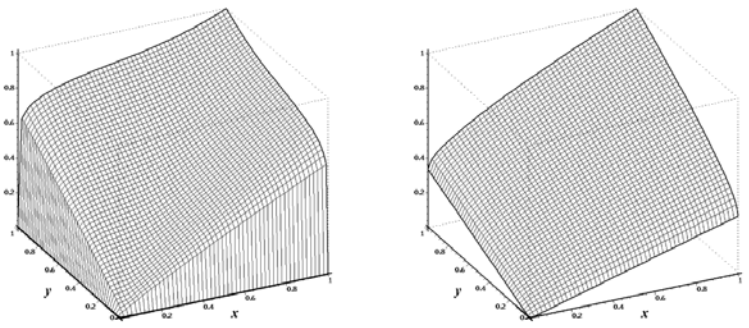


Fig. 2.16. 3D plots of Bonferroni mean $B^{5,0.5}$ and the Heronian mean.

Note 2.44. Generalized logarithmic means are also called Stolarsky means, sometimes L^p is called L^{p+1}.

Note 2.45. The generalized logarithmic mean is symmetric. The limiting cases $x = 0$ depend on p, although $L^p(0, 0) = 0$.

Special cases

- The function $L^0(x, y)$ is called *identric* mean;
- $L^{-2}(x, y) = G(x, y)$, the geometric mean;
- L^{-1} is called the logarithmic mean;
- $L^{-1/2}$ is the power mean with $p = -1/2$;
- L^1 is the arithmetic mean;
- Only $L^{-1/2}$, L^{-2} and L^1 are quasi-arithmetic means.

Note 2.46. For each value of p the generalized logarithmic mean is strictly increasing in x, y, hence they are aggregation functions.

Note 2.47. Generalized logarithmic means can be extended for n arguments using the mean value theorem for divided differences.

2.4.5 Mean of Bajraktarevic

Definition 2.48 (Mean of Bajraktarevic). *Let* $\mathbf{w}(t) = (w_1(t), \ldots, w_n(t))$ *be a vector of weight functions* $w_i : [0, 1] \to [0, \infty[$*, and let* $g : [0, 1] \to [-\infty, \infty]$ *be a strictly monotone function. The mean of Bajraktarevic is the function*

$$f(\mathbf{x}) = g^{-1}\left(\frac{\sum\limits_{i=1}^{n} w_i(x_i)g(x_i)}{\sum\limits_{i=1}^{n} w_i(x_i)}\right). \tag{2.24}$$

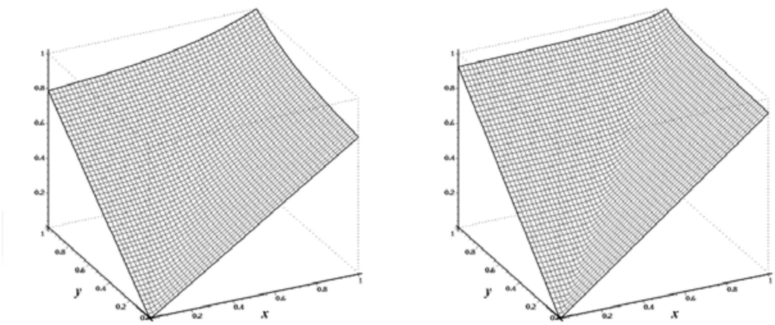


Fig. 2.17. 3D plots of generalized logarithmic means L^{10} and L^{50}.

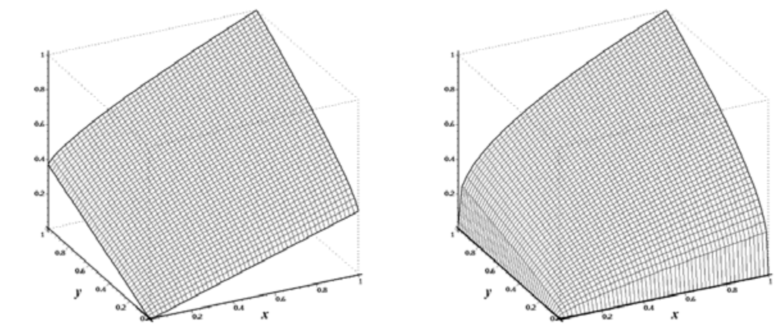


Fig. 2.18. 3D plots of generalized logarithmic means L^0((identric mean) and L^{-1} (logarithmic mean).

The Bajraktarevic mean is also called a *mixture* function [169] when $g(t) = t$. The function g is called the generating function of this mean. If $w_i(t) = w_i$ are constants for all $i = 1, \ldots, n$, it reduces to the quasi-arithmetic mean. The special case of Gini mean $G^{p,q}$ is obtained by taking $w_i(t) = w_i t^q$ and $g(t) = t^{p-q}$ if $p > q$, or $g(t) = \log(t)$ if $p = q$.

Mean of Bajraktarevic is not generally an aggregation function because it fails the monotonicity condition. The following sufficient condition for monotonicity of mixture functions has been established in [169].

Let weight functions $w_i(t) > 0$ be differentiable and monotone non-decreasing, and $g(t) = t$. If $w_i'(t) \le w_i(t)$ for all $t \in [0, 1]$ and all $i = 1, \ldots, n$, then f in (2.24) is monotone non-decreasing (i.e., an aggregation function).

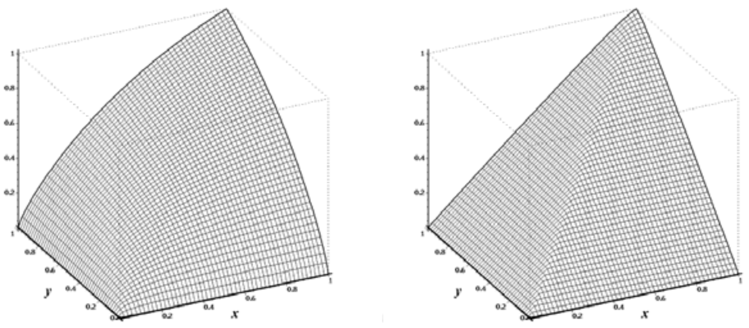


Fig. 2.19. 3D plots of generalized logarithmic means L^{-5} and L^{-100}.

2.5 Ordered Weighted Averaging

2.5.1 Definitions

Ordered weighted averaging functions (OWA) also belong to the class of averaging aggregation functions. They differ to the weighted arithmetic means in that the weights are associated not with the particular inputs, but with their magnitude. In some applications, all inputs are equivalent, and the importance of an input is determined by its value. For example, when a robot navigates obstacles using several sensors, the largest input (the closest obstacle) is the most important. OWA are symmetric aggregation functions that allocate weights according to the input value. Thus OWA can emphasize the largest, the smallest or mid-range inputs. They have been introduced by Yager [263] and have become very popular in the fuzzy sets community.

We recall the notation $\mathbf{x}_\searrow$, which denotes the vector obtained from $\mathbf{x}$ by arranging its components in *non-increasing* order $x_{(1)} \geq x_{(2)} \geq \ldots \geq x_{(n)}$.

Definition 2.49 (OWA). *For a given weighting vector* $\mathbf{w}$*,* $w_i \geq 0$*,* $\sum w_i = 1$*, the OWA function is given by*

$$OWA_{\mathbf{w}}(\mathbf{x}) = \sum_{i=1}^{n} w_i x_{(i)} =< \mathbf{w}, \mathbf{x}_\searrow > .$$

Calculation of the value of an OWA function involves using a `sort()` operation.

Special cases

- If all weights are equal, $w_i = \frac{1}{n}$, OWA becomes the arithmetic mean $OWA_{\mathbf{w}}(\mathbf{x}) = M(\mathbf{x})$;

- If $\mathbf{w} = (1, 0, \ldots, 0)$, then $OWA_{\mathbf{w}}(\mathbf{x}) = \max(\mathbf{x})$;
- If $\mathbf{w} = (0, \ldots, 0, 1)$, then $OWA_{\mathbf{w}}(\mathbf{x}) = \min(\mathbf{x})$;
- If $\mathbf{w} = (\alpha, 0, \ldots, 0, 1-\alpha)$, then OWA becomes the **Hurwizc aggregation function**, $OWA_{\mathbf{w}}(\mathbf{x}) = \alpha \max(\mathbf{x}) + (1-\alpha)\min(\mathbf{x})$;
- If $w_i = 0$ for all i except the k-th, and $w_k = 1$, then OWA becomes k-th order statistic, $OWA_{\mathbf{w}}(\mathbf{x}) = x_{(k)}$.

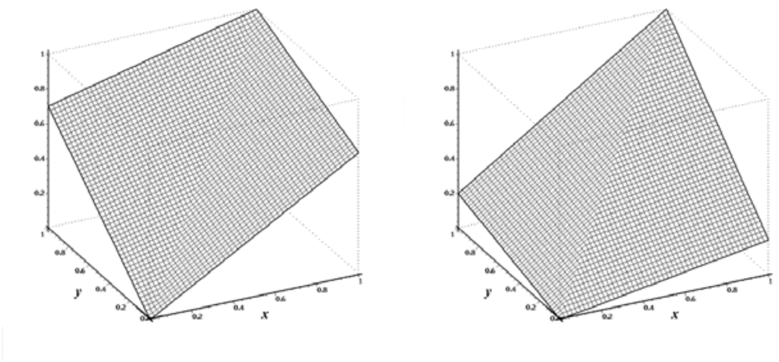


Fig. 2.20. 3D plots of OWA functions $OWA_{(0.7,0.3)}$ and $OWA_{(0.2,0.8)}$.

Definition 2.50 (Reverse OWA). *Given an OWA function $OWA_{\mathbf{w}}$, the reverse OWA is $OWA_{\mathbf{w}_d}$ with the weighting vector $\mathbf{w}_d = (w_n, w_{n-1}, \ldots, w_1)$.*

2.5.2 Main properties

- As with all averaging aggregation functions, OWA are non-decreasing (strictly increasing if all weights are positive) and idempotent;
- The dual of an OWA function is the *reverse* OWA, with the vector of weights $\mathbf{w}_d = (w_n, w_{n-1}, \ldots, w_1)$.
- OWA functions are continuous, symmetric, homogeneous and shift-invariant;
- OWA functions do not have neutral or absorbing elements, except for the special cases min and max;
- The OWA functions are special cases of the Choquet integral (see Section 2.6) with respect to symmetric fuzzy measures.

Orness measure

The general expression for the measure of orness, given in (2.1), translates into the following simple formula

$$orness(OWA_{\mathbf{w}}) = \sum_{i=1}^{n} w_i \frac{n-i}{n-1} = OWA_{\mathbf{w}}(1, \frac{n-2}{n-1}, \ldots, \frac{1}{n-1}, 0). \quad (2.25)$$

Here is a list of additional properties involving the orness value.

- The orness of OWA and its dual are related by

$$orness(OWA_{\mathbf{w}}) = 1 - orness(OWA_{\mathbf{w}_d}).$$

- An OWA function is self-dual if and only if $orness(OWA_{\mathbf{w}}) = \frac{1}{2}$.
- In the special cases $orness(\max) = 1$, $orness(\min) = 0$, and $orness(M) = \frac{1}{2}$. Furthermore, the orness of OWA is 1 only if it is the max function and 0 only if it is the min function. However orness can be $\frac{1}{2}$ for an OWA different from the arithmetic mean, which is nevertheless self-dual.
- If the weighting vector is non-decreasing, i.e., $w_i \le w_{i+1}, i = 1, \ldots, n-1$, then $orness(OWA_{\mathbf{w}}) \in [\frac{1}{2}, 1]$. If the weighting vector is non-increasing, then $orness(OWA_{\mathbf{w}}) \in [0, \frac{1}{2}]$.
- If two OWA functions with weighing vectors $\mathbf{w}_1$, $\mathbf{w}_2$ have their respective orness values O_1, O_2, and if $\mathbf{w}_3 = a\mathbf{w}_1 + (1-a)\mathbf{w}_2$, $a \in [0, 1]$, then OWA function with the weighting vector $\mathbf{w}_3$ has orness value [3]

$$orness(OWA_{\mathbf{w}_3}) = aO_1 + (1-a)O_2.$$

Note 2.51. Of course, to determine an OWA weighting vector with the desired orness value, one can use many different combinations of $\mathbf{w}_1$, $\mathbf{w}_2$, which all result in different $\mathbf{w}_3$ but with the same orness value.

Example 2.52. The measure of orness for some special weighting vectors has been precalculated in [3]

$$w_i = \frac{1}{n} \sum_{j=i}^{n} \frac{1}{j}, \quad orness(OWA_{\mathbf{w}}) = \frac{3}{4},$$

$$w_i = \frac{2(n+1-i)}{n(n+1)}, \quad orness(OWA_{\mathbf{w}}) = \frac{2}{3}.$$

Note 2.53. The classes of recursive and iterative OWA functions, which have the same orness value for any given n, was investigated in [243].

Entropy

We recall the definition of weights dispersion (entropy), Definition 2.37, p. 57

$$Disp(\mathbf{w}) = -\sum_{i=1}^{n} w_i \log w_i.$$

It measures the degree to which all the information (i.e., all the inputs) is used in the aggregation process. The entropy is used to define the weights with the maximal entropy (functions called MEOWA), subject to a predefined orness value (details are in Section 2.5.5).

- If the orness is not specified, the maximum of $Disp$ is achieved at $w_i = \frac{1}{n}$, i.e., the arithmetic mean, and $Disp(\frac{1}{n}, \ldots, \frac{1}{n}) = \log n$.
- The minimum value of $Disp$, 0, is achieved if and only if $w_i = 0$, $i \neq k$, and $w_k = 1$, i.e., the order statistic, see Section 2.8.2.
- The entropy of an OWA and its dual (reverse OWA) coincide, $Disp(\mathbf{w}) = Disp(\mathbf{w}_d)$.

Similarly to the case of weighted quasi-arithmetic means, weighting triangles (see Section 2.3.5) should be used if one needs to work with families of OWAs (i.e., OWA extended aggregation functions in the sense of Definition 1.6). We also note that there are other types of entropy (e.g., Rényi entropy) used to quantify weights dispersion, see, e.g., [266]. One such measure of dispersion was presented in [241] and is calculated as

$$\rho(\mathbf{w}) = \frac{\sum\limits_{i=1}^{n} \frac{i-1}{n-1}\left(w_{(i)} - w_{(i+1)}\right)}{\sum\limits_{i=1}^{n}\left(w_{(i)} - w_{(i+1)}\right)} = \frac{1}{n-1}\frac{1-w_{(1)}}{w_{(1)}},$$

where $w_{(i)}$ denotes the i-th largest weight. A related measure of weights dispersion is $\tilde{\rho}(\mathbf{w}) = 1 - w_{(1)}$ [265]. Another useful measure is weights variance [101], see Eq.(2.38) on p. 79.

2.5.3 Other types of OWA functions

Weighted OWA

The weights in weighted means and OWA functions represent different aspects. In weighted means w_i reflects the importance of the i-th input, whereas in OWA it reflects the importance of the i-th largest input. In [232] Torra proposed a generalization of both weighted means and OWA, called weighted OWA (WOWA). This aggregation function has two sets of weights $\mathbf{w}$, $\mathbf{p}$. Vector $\mathbf{p}$ plays the same role as the weighting vector in weighted means, and $\mathbf{w}$ plays the role of the weighting vector in OWA functions.

Consider the following motivation. A robot needs to combine information coming from n different sensors, which provide distances to the obstacles. The reliability of the sensors is known (i.e., we have weights $\mathbf{p}$). However, independent of their reliability, the distances to the nearest obstacles are more important, so irrespective of the reliability of each sensor, their inputs are also weighted according to their numerical value, hence we have another weighting vector $\mathbf{w}$. Thus both factors, the size of the inputs and the reliability of the inputs, need to be taken into account. WOWA provides exactly this type of aggregation function.

WOWA function becomes the weighted arithmetic mean if $w_i = \frac{1}{n}, i = 1, \ldots, n$, and becomes the usual OWA if $p_i = \frac{1}{n}, i = 1, \ldots, n$.

Definition 2.54 (Weighted OWA). *Let $\mathbf{w}, \mathbf{p}$ be two weighting vectors, $w_i, p_i \geq 0$, $\sum w_i = \sum p_i = 1$. The following function is called Weighted OWA function*

$$WOWA_{\mathbf{w},\mathbf{p}}(\mathbf{x}) = \sum_{i=1}^{n} u_i x_{(i)},$$

where $x_{(i)}$ is the i-th largest component of $\mathbf{x}$, and the weights u_i are defined as

$$u_i = g\left(\sum_{j \in H_i} p_j\right) - g\left(\sum_{j \in H_{i-1}} p_j\right),$$

where the set $H_i = \{j | x_j \geq x_i\}$ is the set of indices of i largest elements of $\mathbf{x}$, and g is a monotone non-decreasing function with two properties:

1. *$g(i/n) = \sum_{j \leq i} w_j, i = 0, \ldots, n$ (of course $g(0) = 0$);*
2. *g is linear if the points $(i/n, \sum_{j \leq i} w_j)$ lie on a straight line.*

Thus computation of WOWA involves a very similar procedure as that of OWA (i.e., sorting components of $\mathbf{x}$ and then computing their weighted sum), but the weights u_i are defined by using both vectors $\mathbf{w}, \mathbf{p}$, a special monotone function g, and depend on the components of $\mathbf{x}$ as well. One can see WOWA as an OWA function with the weights $\mathbf{u}$. Let us list some of the properties of WOWA.

- First, the weighting vector $\mathbf{u}$ satisfies $u_i \geq 0$, $\sum u_i = 1$.
- If $w_i = \frac{1}{n}$, then $WOWA_{\mathbf{w},\mathbf{p}}(\mathbf{x}) = M_{\mathbf{p}}(\mathbf{x})$, the weighted arithmetic mean.
- If $p_i = \frac{1}{n}$, $WOWA_{\mathbf{w},\mathbf{p}}(\mathbf{x}) = OWA_{\mathbf{w}}(\mathbf{x})$.
- WOWA is an idempotent aggregation function.

Of course, the weights $\mathbf{u}$ also depend on the generating function g. This function can be chosen as a linear spline (i.e., a broken line interpolant), interpolating the points $(i/n, \sum_{j \leq i} w_j)$ (in which case it automatically becomes a linear function if these points are on a straight line), or as a monotone quadratic spline, as was suggested in [232, 234], see also [14] where Schumaker's quadratic spline algorithm was used [219], which automatically satisfies the straight line condition when needed.

It turns out that WOWA belongs to a more general class of Choquet integral based aggregation functions, discussed in Section 2.6, with respect to *distorted probabilities*, see Definition 2.113 [193, 233, 237]. It is a piecewise linear function whose linear segments are defined on the simplicial partition of the unit cube $[0,1]^n$: $\mathcal{S}_i = \{\mathbf{x} \in [0,1]^n | x_{p(j)} \geq x_{p(j+1)}\}$, where p is a permutation of the set $\{1, \ldots, n\}$. Note that there are exactly $n!$ possible permutations, the union of all $\mathcal{S}_i$ is $[0,1]^n$, and the intersection of the interiors of $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset, i \neq j$.

Neat OWA

OWA functions have been generalized to functions whose weights depend on the aggregated inputs.

Definition 2.55 (Neat OWA). *An OWA function whose weights are defined by*

$$w_i = \frac{x_{(i)}^p}{\sum\limits_{i=1}^n x_{(i)}^p},$$

with $p \in]-\infty, \infty[$ *is called a neat OWA.*

Note 2.56. Neat OWA functions are counter-harmonic means (see p. 63). We remind that they are not monotone (hence not aggregation functions).

2.5.4 Generalized OWA

Similarly to quasi-arithmetic means (Section 2.3), OWA functions have been generalized with the help of generating functions $g : [0,1] \to [-\infty, \infty]$ as

---

Definition 2.57 (Generalized OWA). *Let* $g : [0,1] \to [-\infty, \infty]$ *be a continuous strictly monotone function and let* $\mathbf{w}$ *be a weighting vector. The function*

$$GenOWA_{\mathbf{w},g}(\mathbf{x}) = g^{-1}\left(\sum_{i=1}^n w_i g(x_{(i)})\right) \qquad (2.26)$$

is called a generalized OWA (also known as ordered weighted quasi-arithmetic mean [43]). As for OWA, $x_{(i)}$ *denotes the* i*-th largest value of* $\mathbf{x}$*.*

Special cases

Ordered weighted geometric function was studied in [125, 260].

Definition 2.58 (Ordered Weighted Geometric function (OWG)). *For a given weighting vector* $\mathbf{w}$*, the OWG function is*

$$OWG_{\mathbf{w}}(\mathbf{x}) = \prod_{i=1}^n x_{(i)}^{w_i}. \qquad (2.27)$$

Note 2.59. Similarly to the weighted geometric mean, OWG is a special case of (2.26) with the generating function $g = \log$.

Definition 2.60 (Ordered Weighted Harmonic function (OWH)). *For a given weighting vector* $\mathbf{w}$*, the OWH function is*

$$OWH_{\mathbf{w}}(\mathbf{x}) = \left(\sum_{i=1}^{n} \frac{w_i}{x_{(i)}}\right)^{-1}. \tag{2.28}$$

A large family of generalized OWA functions is based on power functions, similar to weighted power means [271]. Let g_r denote the family of power functions

$$g_r(t) = \begin{cases} t^r, & \text{if } r \neq 0, \\ \log(t), & \text{if } r = 0. \end{cases}$$

Definition 2.61 (Power-based generalized OWA). *For a given weighting vector* $\mathbf{w}$*, and a value* $r \in \Re$*, the function*

$$GenOWA_{\mathbf{w},[r]}(\mathbf{x}) = \left(\sum_{i=1}^{n} w_i x_{(i)}^r\right)^{1/r}, \tag{2.29}$$

if $r \neq 0$*, and* $GenOWA_{\mathbf{w},[r]}(\mathbf{x}) = OWG_{\mathbf{w}}(\mathbf{x})$ *if* $r = 0$*, is called a power-based generalized OWA .*

Of course, both OWG and OWH functions are special cases of power-based OWA with $r = 0$ and $r = -1$ respectively. The usual OWA corresponds to $r = 1$. Another special case is that of quadratic OWA, $r = 2$, given by

$$OWQ_{\mathbf{w}}(\mathbf{x}) = \sqrt{\sum_{i=1}^{n} w_i x_{(i)}^2}.$$

Other generating functions can also be used to define generalized OWA functions.

Example 2.62 (Trigonometric OWA). Let $g_1(t) = \sin(\frac{\pi}{2}t)$, $g_2(t) = \cos(\frac{\pi}{2}t)$, and $g_3(t) = \tan(\frac{\pi}{2}t)$ be the generating functions. The trigonometric OWA functions are the functions

$$OWAS_{\mathbf{w}}(\mathbf{x}) = \frac{2}{\pi}\arcsin\Big(\sum_{i=1}^{n} w_i \sin\Big(\frac{\pi}{2}x_{(i)}\Big)\Big),$$

$$OWAC_{\mathbf{w}}(\mathbf{x}) = \frac{2}{\pi}\arccos\Big(\sum_{i=1}^{n} w_i \cos\Big(\frac{\pi}{2}x_{(i)}\Big)\Big), \text{ and}$$

$$OWAT_{\mathbf{w}}(\mathbf{x}) = \frac{2}{\pi}\arctan\Big(\sum_{i=1}^{n} w_i \tan\Big(\frac{\pi}{2}x_{(i)}\Big)\Big).$$

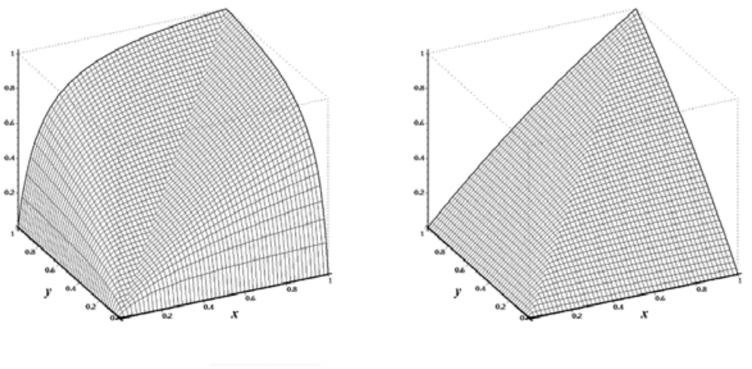


Fig. 2.21. 3D plots of OWH functions $OWH_{(0.9,0.1)}$ and $OWH_{(0.2,0.8)}$.

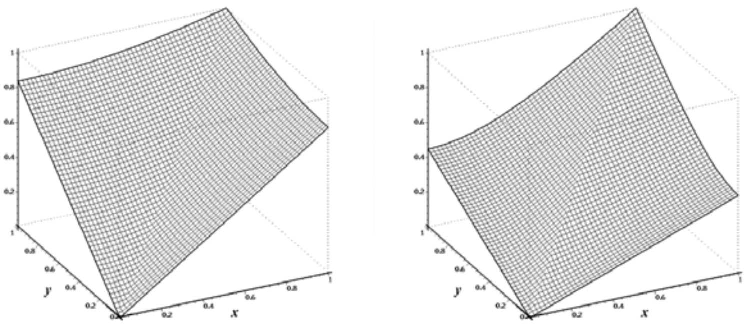


Fig. 2.22. 3D plots of quadratic OWA functions $OWQ_{(0.7,0.3)}$ and $OWQ_{(0.2,0.8)}$.

Example 2.63 (Exponential OWA). Let the generating function be

$$g(t) = \begin{cases} \gamma^t, \text{ if } \gamma \neq 1, \\ t, \quad \text{if } \gamma = 1. \end{cases}$$

The exponential OWA is the function

$$OWAE_{\mathbf{w},\gamma}(\mathbf{x}) = \begin{cases} \log_\gamma \left(\sum_{i=1}^n w_i \gamma^{x_{(i)}} \right), \text{ if } \gamma \neq 1, \\ OWA_{\mathbf{w}}(\mathbf{x}), \qquad\qquad\quad \text{if } \gamma = 1. \end{cases}$$

Example 2.64 (Radical OWA). Let $\gamma > 0$, $\gamma \neq 1$, and let the generating function be

$$g(t) = \gamma^{1/t}.$$

The radical OWA is the function

$$OWAR_{\mathbf{w},\gamma}(\mathbf{x}) = \left(\log_\gamma(\sum_{i=1}^{n} w_i \gamma^{1/x_{(i)}})\right)^{-1}.$$

3D plots of some generalized OWA functions are presented on Figures 2.22-2.24.

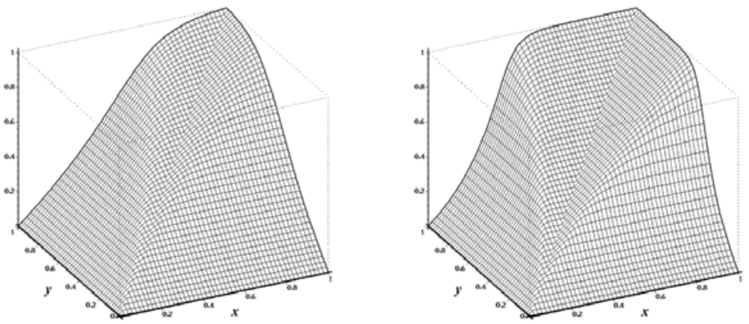


Fig. 2.23. 3D plots of radical OWA functions $OWAR_{(0.9,0.1),100}$ and $OWAR_{(0.999,0.001),100}$.

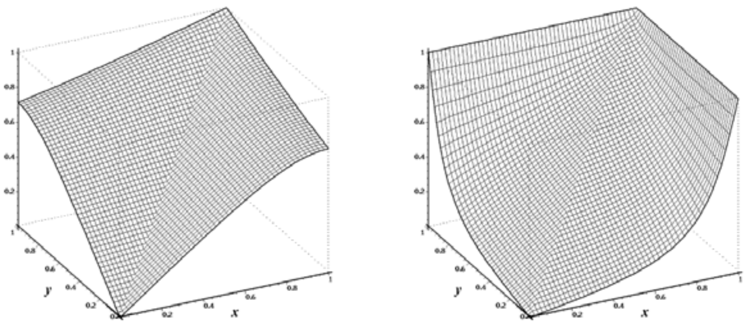


Fig. 2.24. 3D plots of trigonometric OWA functions $OWAS_{(0.9,0.1)}$ and $OWAT_{(0.2,0.8)}$.

2.5.5 How to choose weights in OWA

Methods based on data

The problem of identification of weights of OWA functions was studied by several authors [93, 100, 259, 276]. A common feature of all methods is to eliminate nonlinearity due to reordering of the components of $\mathbf{x}$ by restricting the domain of this function to the simplex $S \subset [0,1]^n$ defined by the inequalities $x_1 \geq x_2 \geq \ldots \geq x_n$. On that domain OWA function is a linear function (it coincides with the arithmetic mean). Once the coefficients of this function are found, OWA function can be computed on the whole $[0,1]^n$ by using its symmetry. Algorithmically, it amounts to using an auxiliary data set $\{(\mathbf{z}_k, y_k)\}$, where vectors $\mathbf{z}_k = \mathbf{x}_{k\searrow}$. Thus identification of weights of OWA functions is a very similar problem to identification of weights of arithmetic means in Section 2.3.7. Depending on whether we use least squares or least absolute deviation criterion, we solve it by using either quadratic or linear programming techniques. In the first case we have the problem

$$\begin{aligned} \min \quad & \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i z_{ik} - y_k \right)^2 \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n. \end{aligned} \tag{2.30}$$

In the second case we have

$$\begin{aligned} \min \quad & \sum_{k=1}^{K} \left| \sum_{i=1}^{n} w_i z_{ik} - y_k \right| \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n, \end{aligned} \tag{2.31}$$

which converts to a linear programming problem, see Appendix A.2.

Filev and Yager [93] proposed a nonlinear change in variables to obtain an unrestricted minimization problem, which they propose to solve using nonlinear local optimization methods. Unfortunately the resulting nonlinear optimization problem is difficult due to a large number of local minimizers, and the traditional optimization methods are stuck in the local minima.

The approach relying on quadratic programming was used in [17, 20, 235, 236, 275], and it was shown to be numerically efficient and stable with respect to rank deficiency (e.g., when $K < n$, or the data are linearly dependent).

Often an additional requirement is imposed: the desired value of the measure of orness $orness(f) = \alpha \in [0,1]$. This requirement is easily incorporated into a QP or LP problem as an additional linear equality constraint, namely

$$\sum_{i=1}^{n} w_i \frac{n-i}{n-1} = \alpha.$$

Preservation of ordering of the outputs

We may also require that the ordering of the outputs is preserved, i.e., if $y_j \leq y_k$ then we expect $f(\mathbf{x}_j) \leq f(\mathbf{x}_k)$ (see Section 1.6, p. 34). We arrange the data, so that the outputs are in non-decreasing order, $y_k \leq y_{k+1}, k = 1, \ldots, K-1$. Then we define the additional linear constraints

$$< \mathbf{z}_{k+1} - \mathbf{z}_k, \mathbf{w} >= \sum_{i=1}^{n} w_i(z_{i,k+1} - z_{ik}) \geq 0,$$

$k = 1, \ldots, K-1$. We add the above constraints to problem (2.30) or (2.31) and solve it. The addition of an extra $K-1$ constraints neither changes the structure of the optimization problem, nor does it drastically affect its complexity.

Methods based on a measure of dispersion

Maximum entropy OWA

A different approach to choosing OWA weights was proposed in [199] and followed in [100]. It does not use any empirical data, but various measures of weight entropy or dispersion. The measure of weights dispersion (see Definition 2.7 on p. 57, also see p. 70) is defined as

$$Disp(\mathbf{w}) = -\sum_{i=1}^{n} w_i \log w_i, \tag{2.32}$$

The idea is to choose for a given n such a vector of weights that maximizes the dispersion $Disp(\mathbf{w})$.

It is formulated as an optimization problem

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} w_i \log w_i \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1, \\ & \sum_{i=1}^{n} w_i \frac{n-i}{n-1} = \alpha, \\ & w_i \geq 0, i = 1, \ldots, n. \end{aligned} \tag{2.33}$$

The solution is provided in [100] and is called Maximum Entropy OWA (MEOWA). Using the method of Lagrange multipliers, the authors obtain the following expressions for w_i:

$$w_i = (w_1^{n-i} w_n^{i-1})^{\frac{1}{n-1}}, i = 2, \ldots, n-1, \tag{2.34}$$

$$w_n = \frac{((n-1)\alpha - n)w_1 + 1}{(n-1)\alpha + 1 - nw_1},$$

and w_1 being the unique solution to the equation

$$w_1[(n-1)\alpha + 1 - nw_1]^n = ((n-1)\alpha)^{n-1}[((n-1)\alpha - n)w_1 + 1] \quad (2.35)$$

on the interval $(0, \frac{1}{n})$.

Note 2.65. For $n = 3$, we obtain $w_2 = \sqrt{w_1 w_3}$ independently of the value of α.

A different representation of the same solution was given in [51]. Let t be the (unique) positive solution to the equation

$$dt^{n-1} + (d+1)t^{n-2} + \ldots + (d+n-2)t + (d+n-1) = 0, \quad (2.36)$$

with $d = -\alpha(n-1)$. Then the MEOWA weights are identified from

$$w_i = \frac{t^i}{T}, \ i = 1, \ldots, n, \quad \text{where } T = \sum_{j=1}^{n} t^j. \quad (2.37)$$

Note 2.66. It is not difficult to check that both (2.34) and (2.37) represent the same set of weights, noting that $t = \sqrt[n-1]{\frac{w_n}{w_1}} = -\frac{1-d-nw_1}{d}$, or $w_1 = \frac{1+td-d}{n}$, and that substituting w_1 into (2.35) yields

$$1 - t^n = \frac{n(1-t)}{1-d(1-t)},$$

which translates into

$$\frac{1-t^n}{1-t} - d(1-t^n) - n = 0,$$

and then into

$$dt^n + t^{n-1} + t^{n-2} + \ldots + t + (1-d-n) = 0.$$

After factoring out $(t-1)$ we obtain (2.36).

Minimum variance OWA

Another popular characteristic of weighting vector is weights variance, defined as [101]

$$D^2(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}(w_i - M(\mathbf{w}))^2 = \frac{1}{n}\sum_{i=1}^{n} w_i^2 - \frac{1}{n^2}, \quad (2.38)$$

where $M(\mathbf{w})$ is the arithmetic mean of $\mathbf{w}$.

Here one minimizes $D^2(\mathbf{w})$ subject to given orness measure. The resulting OWA function is called Minumum Variance OWA (MVOWA). Since adding a constant to the objective function does not change the minimizer, this is equivalent to the problem

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} w_i^2 \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i \tfrac{n-i}{n-1} = \alpha, \\ & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n. \end{aligned} \tag{2.39}$$

For $\alpha = \frac{1}{2}$ the optimal solution is always $w_j = \frac{1}{n}, j = 1, \ldots, n$. It is also worth noting that the optimal solution to (2.39) for $\alpha > \frac{1}{2}$, $\mathbf{w}^*$, is related to the optimal solution for $\alpha < \frac{1}{2}$, $\mathbf{w}$, by $w_i^* = w_{n-i+1}$, i.e., it gives the reverse OWA. Thus it is sufficient to establish the optimal solution in the case $\alpha < \frac{1}{2}$.

The optimal solution [101, 157] for $\alpha < \frac{1}{2}$ is given as the vector $\mathbf{w} = (0, 0, \ldots, 0, w_r, \ldots, w_n)$, i.e., $w_j = 0$ if $j < r$, and

$$w_r = \frac{6(n-1)\alpha - 2(n-r-1)}{(n-r+1)(n-r+2)},$$

$$w_n = \frac{2(2n-2r+1) - 6(n-1)\alpha)}{(n-r+1)(n-r+2)},$$

and

$$w_j = w_r + \frac{j-r}{n-r}(w_n - w_r), \; r < j < n.$$

The index r depends on the value of α, and is found from the inequalities

$$n - 3(n-1)\alpha - 1 < r \leq n - 3(n-1)\alpha.$$

Recently it was established [157] that the solution to the minimum variance OWA weights problem is equivalent to that of minimax disparity [251], i.e., the solution to

$$\begin{aligned} \min \quad & \left\{ \max_{i=1,\ldots,n-1} |w_i - w_{i-1}| \right\} \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i \tfrac{n-i}{n-1} = \alpha, \\ & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n. \end{aligned} \tag{2.40}$$

We reiterate that the weights of OWA functions obtained as solutions to maximum entropy or minimum variance problems are fixed for any given n and orness measure, and can be precomputed. However, both criteria are also useful for data driven weights identification (in Section 2.5.5), if there are multiple optimal solutions. Then the solution maximizing $Disp(\mathbf{w})$ or minimizing $D(\mathbf{w})$ is chosen. Torra [235] proposes to solve an auxiliary univariate optimization problem to maximize weights dispersion, subject to a given value of (2.32). On the other hand, one can fit the orness value α of MEOWA or

MVOWA to empirical data, using a univariate nonlinear optimization method, in which at each iteration the vector $\mathbf{w}$ is computed using analytical solutions to problems (2.33) and (2.39).

Furthermore, it is possible to include both criteria directly into problem (2.30). It is especially convenient for the minimum variance criterion, as it yields a modified quadratic programming problem

$$\begin{aligned} \min \; & \sum_{k=1}^{K}\left(\sum_{i=1}^{n} w_i z_{ik} - y_k\right)^2 + \lambda \sum_{i=1}^{n} w_i^2 \\ \text{s.t.} \; & \sum_{i=1}^{n} w_i \tfrac{n-i}{n-1} = \alpha, \\ & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n, \end{aligned} \tag{2.41}$$

where $\lambda \geq 0$ is a user-specified parameter controlling the balance between the criterion of fitting the data and that of obtaining minimum variance weights.

Methods based on weight generating functions

Yager [264, 267] has proposed to use monotone continuous functions $Q : [0,1] \to [0,1], Q(0) = 0, Q(1) = 1$, called Basic Unit-interval Monotone (BUM) functions, or Regular Increasing Monotone (RIM) quantifiers [264]. These functions generate OWA weights for any n using (see Section 2.3.5)

$$w_i = Q\left(\frac{i}{n}\right) - Q\left(\frac{i-1}{n}\right). \tag{2.42}$$

RIM quantifiers are fuzzy linguistic quantifiers [13] that express the concept of fuzzy majority. Yager defined such quantifiers for fuzzy sets "for all", "there exists", "identity", "most", "at least half", "as many as possible" as follows.

- "for all": $Q_{forall}(t) = 0$ for all $t \in [0, 1)$ and $Q_{forall}(1) = 1$.
- "there exists": $Q_{exists}(t) = 1$ for all $t \in (0, 1]$ and $Q_{exists}(0) = 0$.
- "identity": $Q_{Id}(t) = t$.

Other mentioned quantifiers are expressed by

$$Q_{a,b}(t) = \begin{cases} 0, & \text{if } t \leq a, \\ \frac{t-a}{b-a} & \text{if } a < t < b, \\ 1, & \text{if } t \geq b. \end{cases} \tag{2.43}$$

Then we can choose pairs $(a, b) = (0.3, 0.8)$ for "most", $(a, b) = (0, 0.5)$ for "at least half" and $(a, b) = (0.5, 1)$ for "as many as possible".

Calculation of weights results in the following OWA:

[13] I.e., Q is a monotone increasing function $[0, 1] \to [0, 1]$, $Q(0) = 0, Q(1) = 1$ whose value $Q(t)$ represents the degree to which t satisfies the fuzzy concept represented by the quantifier.

- "for all": $\mathbf{w} = (0, 0, \ldots, 0, 1)$, $OWA_{\mathbf{w}} = \min$.
- "there exists": $\mathbf{w} = (1, 0, 0, \ldots, 0)$, $OWA_{\mathbf{w}} = \max$.
- "identity": $\mathbf{w} = (\frac{1}{n}, \ldots, \frac{1}{n})$, $OWA_{\mathbf{w}} = M$.

Example 2.67. Consider linguistic quantifier "most", given by (2.43) with $(a, b) = (0.3, 0.8)$ and $n = 5$. The weighting vector is then $(0, 0.2, 0.4, 0.4, 0)$.

Weight generating functions are applied to generate weights of both quasi-arithmetic means and OWA functions. They allow one to compute the degree of orness of an OWA function in the limiting case

$$\lim_{n \to \infty} orness(f_n) = orness(Q) = \int_0^1 Q(t)dt.$$

Entropy and other characteristics can also be computed based on Q, see [244].

Yager [274] has proposed using generating, or stress functions (see also [156]), defined by

---

Definition 2.68 (Generating function of RIM quantifiers). *Let $q : [0,1] \to [0, \infty]$ be an (integrable) function. It is a generating function of RIM quantifier Q, if*

$$Q(t) = \frac{1}{K} \int_0^t q(u)du,$$

where $K = \int_0^1 q(u)du$ is the normalization constant. The normalized generating function will be referred to as $\tilde{q}(t) = \frac{q(t)}{K}$.

Note 2.69. The generating function has the properties of a density function (e.g., a probability distribution density, although Q is not necessarily interpreted as a probability). If Q is differentiable, we may put $q(t) = Q'(t)$. Of course, for a given Q, if a generating function exists, it is not unique.

Note 2.70. In general, Q needs not be continuous to have a generating function. For example, it may be generated by Dirac's delta function [14]

$$\delta(t) = \begin{cases} \infty, & \text{if } t = 0, \\ 0 & \text{otherwise,} \end{cases}$$

constrained by $\int_{-\infty}^{\infty} \delta(t)dt = 1$.

By using the generating function we generate the weights as

$$\tilde{w}_i = \frac{1}{K} q\left(\frac{i}{n}\right) \frac{1}{n}.$$

Note that these weights provide an approximation to the weights generated by (2.42), and that they do not necessarily sum to one. To ensure the latter, we shall use the weights

[14] This is an informal definition. The proper definition involves the concepts of distributions and measures, see, e.g., [213].

$$w_i = \frac{\frac{1}{K} q\left(\frac{i}{n}\right)\frac{1}{n}}{\sum_{j=1}^{n} \frac{1}{K} q\left(\frac{j}{n}\right)\frac{1}{n}} = \frac{q\left(\frac{i}{n}\right)}{\sum_{j=1}^{n} q\left(\frac{j}{n}\right)}. \tag{2.44}$$

Eq. (2.44) provides an alternative method for OWA weight generation, independent of Q, while at the same time it gives an approximation to the weights provided by (2.42). Various interpretations of generating functions are provided in [274], from which we quote just a few examples.

Example 2.71.

- A constant generating function $q(t) = 1$ generates weights $w_i = \frac{1}{n}$, i.e., the arithmetic mean.
- Constant in range function $q(t) = 1$ for $t \le \beta$ and 0 otherwise, emphasizes the larger arguments, and generates the weights $w_i = \frac{1}{r}, i = 1, \ldots, r$ and $w_i = 0, i = r+1, \ldots, n$, where r is the largest integer less or equal βn.
- Generating function $q(t) = 1$, for $\alpha \le t \le \beta$ and 0 otherwise, emphasizes the "middle" arguments, and generates the weights $w_i = \frac{1}{p-r}, i = r + 1, \ldots, p$ and 0 otherwise, with (for simplicity) $\alpha n = r$ and $\beta n = p$.
- Generating function with two tails $q(t) = 1$ if $t \in [0, \alpha]$ or $t \in [\beta, 1]$ and 0 otherwise, emphasizes both large and small arguments and yields $w_i = \frac{1}{r_1 + r_2}$ for $i = 1, \ldots, r_1$ and $i = n + 1 - r_2, \ldots, n$, and $w_i = 0, i = r_1 + 1, \ldots, n - r_2$, with $r_1 = \alpha n$, $r_2 = \beta n$ integers.
- Linear stress function $q(t) = t$ generates weights $w_i = \frac{i}{\sum_{j=1}^{n} i} = \frac{2i}{n(n+1)}$, which gives orness value $\frac{1}{3}$, compare to Example 2.52. It emphasizes smaller arguments.

Of course, by using the same approach (i.e., $Q(t)$ or $q(t)$) one can generate the weights of generalized OWA and weighted quasi-arithmetic means. However the interpretation and the limiting cases for the means will be different. For example the weighting vector $\mathbf{w} = (1, 0, \ldots, 0)$ results not in the *max* function, but in the projection to the first coordinate $f(\mathbf{x}) = x_1$.

Fitting weight generating functions

Weight generating functions allow one to compute weighting vectors of OWA and weighted means for any number of arguments, i.e., to obtain extended aggregation functions in the sense of Definition 1.6. This is very convenient when the number of arguments is not known a priori. Next we pose the question as to whether it is possible to learn weight generating functions from empirical data, similarly to determining weighting vectors of aggregation functions of a fixed dimension.

A positive answer was provided in [20, 30]. The method consists in representing a weight generating function with a spline or polynomial, and fitting its coefficients by solving a least squares or least absolute deviation problem subject to a number of linear constraints. Consider a data set

$(\mathbf{x}_k, y_k), k = 1, \ldots, K$, where vectors $\mathbf{x}_k \in [0,1]^{n_k}$ need not have the same dimension (see Table 2.1). This is because we are dealing with an extended aggregation function — a family of n-ary aggregation functions.

Table 2.1. A data set with inputs of varying dimension.

k	n_k	x_1	x_2	x_3	x_4	x_5	y
1	3	x_{11}	x_{21}	x_{31}			y_1
2	2	x_{12}	x_{22}				y_2
3	3	x_{13}	x_{23}	x_{33}			y_3
4	5	x_{14}	x_{24}	x_{34}	x_{44}	x_{54}	y_4
5	4	x_{15}	x_{25}	x_{35}	x_{45}		y_5
$\vdots$							

First, let us use the method of monotone splines, discussed in the Appendix A.1. We write

$$Q(t) = \sum_{j=1}^{J} c_j B_j(t), t \in (0,1) \text{ and } Q(0) = 0, Q(1) = 1,$$

where functions $B_j(t)$ constitute a convenient basis for polynomial splines, [13], in which the condition of monotonicity of Q is expressed as $c_j \geq 0, j = 1, \ldots, J$. We do not require Q to be continuous on $[0,1]$ but only on $]0,1[$. We also have two linear constraints

$$\sum_{j=1}^{J} c_j B_j(0) \geq 0, \quad \sum_{j=1}^{J} c_j B_j(1) \leq 1,$$

which convert to equalities if we want Q to be continuous on $[0,1]$. Next put this expression in (2.42) to get

$$
\begin{aligned}
f(\mathbf{x}_k) &= \sum_{i=1}^{n_k} z_{ik}\left(Q\left(\frac{i}{n_k}\right) - Q\left(\frac{i-1}{n_k}\right)\right) \\
&= \sum_{i=2}^{n_k-1} z_{ik}\left(\sum_{j=1}^{J} c_j\left[B_j\left(\frac{i}{n_k}\right) - B_j\left(\frac{i-1}{n_k}\right)\right]\right) \\
&\quad + z_{1k}\left[\sum_{j=1}^{J} c_j B_j\left(\frac{1}{n_k}\right) - 0\right] + z_{n_k k}\left[1 - \sum_{j=1}^{J} c_j B_j\left(\frac{n_k-1}{n_k}\right)\right] \\
&= \sum_{j=1}^{J} c_j\left(\sum_{i=2}^{n_k-1} z_{ik}\left[B_j\left(\frac{i}{n_k}\right) - B_j\left(\frac{i-1}{n_k}\right)\right]\right. \\
&\quad \left. + z_{1k}B_j\left(\frac{1}{n_k}\right) - z_{n_k k}B_j\left(\frac{n_k-1}{n_k}\right)\right) + z_{n_k k} \\
&= \sum_{j=1}^{J} c_j A_j(\mathbf{x}_k) + z_{n_k k}.
\end{aligned}
$$

The vectors $\mathbf{z}_k$ stand for $\mathbf{x}_k$ when we treat weighted arithmetic means, or $\mathbf{x}_{k\searrow}$ when we deal with OWA functions. The entries $A_j(\mathbf{x}_k)$ are computed from z_{ik} using expression in the brackets. Note that if Q is continuous on $[0,1]$ the expression simplifies to

$$
f(\mathbf{x}_k) = \sum_{j=1}^{J} c_j\left(\sum_{i=1}^{n_k} z_{ik}\left[B_j\left(\frac{i}{n_k}\right) - B_j\left(\frac{i-1}{n_k}\right)\right]\right).
$$

Consider now the least squares approximation of empirical data. We obtain a quadratic programming problem

$$
\begin{aligned}
\text{minimize} \quad & \sum_{k=1}^{K}\left(\sum_{j=1}^{J} c_j A_j(\mathbf{x}_k) + z_{n_k k} - y_k\right)^2 \qquad (2.45)\\
\text{s.t.} \quad & \sum_{j=1}^{J} c_j B_j(0) \geq 0, \quad \sum_{j=1}^{J} c_j B_j(1) \leq 1, \\
& c_j \geq 0.
\end{aligned}
$$

The solution is performed by QP programming methods described in Appendix A.5.

OWA aggregation functions and weighted arithmetic means are special cases of Choquet integral based aggregation functions, described in the next section. Choquet integrals are defined with respect to a fuzzy measure (see Definition 2.75). When the fuzzy measure is additive, Choquet integrals become weighted arithmetic means, and when the fuzzy measure is symmetric, they become OWA functions. There are special classes of fuzzy measures called k-additive measures (see Definition 2.121). We will discuss them in detail in

Section 2.6.3, and in the remainder of this section we will present a method for identifying weight generating functions that correspond to symmetric 2- and 3-additive fuzzy measures. These fuzzy measures lead to OWA functions with special weights distributions.

Proposition 2.72. *[30] A Choquet integral based aggregation function with respect to a symmetric 2-additive fuzzy measure is an OWA function whose weight generating function is given by*

$$Q(t) = at^2 + (1-a)t \text{ for some } a \in [-1,1].$$

Furthermore, such an OWA weighting vector is equidistant (i.e., $w_{i+1} - w_i = const$ for all $i = 1, \dots, n-1$).

A Choquet integral based aggregation function with respect to a symmetric 3-additive fuzzy measure is an OWA function whose weight generating function is given by

$$Q(t) = at^3 + bt^2 + (1-a-b)t \text{ for some } a \in [-2,4],$$

such that

- *if $a \in [-2,1]$ then $b \in [-2a-1, 1-a]$;*
- *if $a \in]1,4]$ then $b \in [-3a/2 - \sqrt{3a(4-a)/4}, -3a/2 + \sqrt{3a(4-a)/4}]$.*

Proposition 2.72 provides two parametric classes of OWA functions that correspond to 2- and 3-additive symmetric fuzzy measures. In these cases, rather than fitting a general monotone non-decreasing function, we fit a quadratic or cubic function, identified by parameters a and b.

Interestingly, in the case of 2-additive symmetric fuzzy measure, we obtain the following formula, a linear combination of OWA and the arithmetic mean

$$f(\mathbf{x}) = aOWA_{\mathbf{w}}(\mathbf{x}) + (1-a)M(\mathbf{x}),$$

with $\mathbf{w} = (\frac{1}{n^2}, \frac{3}{n^2}, \dots, \frac{2n-1}{n^2})$. In this case the solution is explicit, the optimal a is given by [30]

$$a = \max\left\{-1, \min\left\{1, \frac{\sum\limits_{k=1}^{K}(y_k - U_k)V_k}{\sum\limits_{k=1}^{K} V_k^2}\right\}\right\},$$

where

$$U_k = \sum_{i=1}^{n_k} \frac{z_{ik}}{n_k}, \text{ and}$$

$$V_k = \frac{1}{n_k}\sum_{i=1}^{n_k}\left(\frac{2i-1}{n_k} - 1\right) z_{ik}.$$

For 3-additive symmetric fuzzy measures the solution is found numerically by solving a convex optimization problem in the feasible domain D in Proposition 2.72, which is the intersection of a polytope and an ellipse. Details are provided in [30].

2.5.6 Choosing parameters of generalized OWA

Choosing weight generating functions

Consider the case of generalized OWA functions, where a given generating function g is given. As earlier, we have a data set $(\mathbf{x}_k, y_k), k = 1, \ldots, K$, and we are interested in finding the weighting vector $\mathbf{w}$ that fits the data best. When we use the least squares, as discussed on p. 33, we have the following optimization problem

$$\begin{aligned} \min & \sum_{k=1}^{K} \left(g^{-1} \left(\sum_{i=1}^{n} w_i g(z_{ik}) \right) - y_k \right)^2 \\ \text{s.t.} & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n, \end{aligned} \tag{2.46}$$

where $\mathbf{z}_k = \mathbf{x}_{k\searrow}$ (see Section 2.5.5). This problem is converted to a QP problem similarly to the case of weighted quasi-arithmetic means:

$$\begin{aligned} \min & \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i g(z_{ik}) - g(y_k) \right)^2 \\ \text{s.t.} & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n. \end{aligned} \tag{2.47}$$

This is a standard convex QP problem, and the solution methods are discussed in the Appendix A.5. This approach is presented in [20].

If one uses the least absolute deviation (LAD) criterion (p. 33) we obtain a different optimization problem

$$\begin{aligned} \min & \sum_{k=1}^{K} \left| \sum_{i=1}^{n} w_i g(z_{ik}) - g(y_k) \right| \\ \text{s.t.} & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n. \end{aligned} \tag{2.48}$$

This problem is subsequently converted into a linear programming problem as discussed in the Appendix A.2.

As in the case of weighted quasi-arithmetic means, in the presence of multiple optimal solutions, one can use an additional criterion of the dispersion of weights [235].

Preservation of ordering of the outputs

If we require that the ordering of the outputs be preserved, i.e., if $y_j \leq y_k$ then we expect $f(\mathbf{x}_j) \leq f(\mathbf{x}_k)$ (see Section 1.6, p. 34), then we arrange the data, so that the outputs are in a non-decreasing order, $y_k \leq y_{k+1}, k = 1, \ldots, K-1$. Then we define the additional linear constraints

$$< \mathbf{g}(z_{k+1}) - g(\mathbf{z}_k), \mathbf{w} >= \sum_{i=1}^{n} w_i(g(z_{i,k+1}) - g(z_{ik})) \geq 0,$$

$k = 1, \ldots, K-1$. We add the above constraints to problem (2.47) or (2.48) and solve the modified problem.

Choosing generating functions

Consider now the case where the generating function g is also unknown, and hence it has to be found based on the data. We study two cases: a) when g is given algebraically, with one or more unknown parameters to estimate (e.g., $g_r(t) = t^r$, r unknown), and b) when no specific algebraic form of g is given.

In the first case we solve the problem

$$\begin{aligned} \min_{r,\mathbf{w}} \ & \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i g_r(z_{ik}) - g_r(y_k) \right)^2 \\ \text{s.t.} \ & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n, \\ & \text{plus conditions on } r. \end{aligned} \tag{2.49}$$

While this general optimization problem is non-convex and nonlinear (i.e., difficult to solve), we can convert it to a bi-level optimization problem (see Appendix A.5.3)

$$\begin{aligned} \min_{r} \ & \left[\min_{\mathbf{w}} \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i g_r(z_{ik}) - g_r(y_k) \right)^2 \right] \\ \text{s.t.} \ & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n, \\ & \text{plus conditions on } r. \end{aligned} \tag{2.50}$$

The problem at the inner level is the same as (2.47) with a fixed g_r, which is a QP problem. At the outer level we have a global optimization problem with respect to a single parameter r. It is solved by using one of the methods discussed in Appendix A.5.4-A.5.5. We recommend deterministic Pijavski-Shubert method.

Example 2.73. Determine the weights and the generating function of a family of generalized OWA based on the power function, subject to a given measure of orness α. We have $g_r(t) = t^r$, and hence solve bi-level optimization problem

$$\begin{aligned} \min_{r \in [-\infty,\infty]} \ & \left[\min_{\mathbf{w}} \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i z_{ik}^r - y_k^r \right)^2 \right] \\ \text{s.t.} \ & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, i = 1, \ldots, n, \\ & \sum_{i=1}^{n} w_i \left(\frac{n-i}{n-1} \right)^r = \alpha. \end{aligned} \tag{2.51}$$

Of course, for numerical purposes we need to limit the range for r to a finite interval, and treat all the limiting cases $r \to \pm\infty$, $r \to 0$ and $r \to -1$.

A different situation arises when the parametric form of g is not given. The approach proposed in [17, 20] is based on approximation of g with a monotone linear spline, as

$$g(t) = \sum_{j=1}^{J} c_j B_j(t), \tag{2.52}$$

where B_j are appropriately chosen basis functions, and c_j are spline coefficients. The monotonicity of g is ensured by imposing linear restrictions on spline coefficients, in particular non-negativity, as in [13]. Further, since the generating function is defined up to an arbitrary linear transformation, one has to fix a particular g by specifying two interpolation conditions, like $g(a) = 0, g(b) = 1, a, b \in]0, 1[$, and if necessary, properly model asymptotic behavior if $g(0)$ or $g(1)$ are infinite.

After rearranging the terms of the sum, the problem of identification becomes (subject to linear conditions on $\mathbf{c}, \mathbf{w}$)

$$\min_{\mathbf{c},\mathbf{w}} \sum_{k=1}^{K} \left(\sum_{j=1}^{J} c_j \left[\sum_{i=1}^{n} w_i B_j(z_{ik}) - B_j(y_k) \right] \right)^2 . \tag{2.53}$$

For a fixed $\mathbf{c}$ (i.e., fixed g) we have a quadratic programming problem to find $\mathbf{w}$, and for a fixed $\mathbf{w}$, we have a quadratic programming problem to find $\mathbf{c}$. However if we consider both $\mathbf{c}, \mathbf{w}$ as variables, we obtain a difficult global optimization problem. We convert it to a bi-level optimization problem

$$\min_{\mathbf{c}} \min_{\mathbf{w}} \sum_{k=1}^{K} \left(\sum_{j=1}^{J} c_j \left[\sum_{i=1}^{n} w_i B_j(z_{ik}) - B_j(y_k) \right] \right)^2 ,$$

where at the inner level we have a QP problem and at the outer level we have a nonlinear problem with multiple local minima. When the number of spline coefficients J is not very large (< 10), this problem can be efficiently solved by using deterministic global optimization methods from Appendix A.5.5. If the number of variables is small and J is large, then reversing the order of minimization (i.e., using $\min_{\mathbf{w}} \min_{\mathbf{c}}$) is more efficient.

Choosing generating functions and weight generating functions

We remind the definition of Generalized OWA 2.57. Consider the case of generating function $g(t) = t^r$, in which case

$$GenOWA_{\mathbf{w},[r]}(\mathbf{x}) = \left(\sum_{i=1}^{n} w_i z_i^r \right)^{1/r} .$$

Consider first a fixed r. To find a weight generating function $Q(t)$, we first linearize the least squares problem to get

$$\begin{aligned}
&\min_{\mathbf{c}} \sum_{k=1}^{K} \left(\sum_{j=1}^{J} c_j \sum_{i=1}^{n_k} \left[B_j\left(\frac{i}{n_k}\right) - B_j\left(\frac{i-1}{n_k}\right)\right] (z_{ik})^r - y_k^r \right)^2 \\
&\text{s.t.} \qquad \sum_{j=1}^{J} c_j B_j(0) = 0, \quad \sum_{j=1}^{J} c_j B_j(1) = 1, \\
&\qquad\qquad c_j \geq 0.
\end{aligned}$$

This is a standard QP, which differs from (2.45) because z_{ik} and y_k are raised to power r (we considered a simpler case of $Q(t)$ continuous on $[0, 1]$).

Now, if the parameter r is also unknown, we determine it from data by setting a bi-level optimization problem

$$\begin{aligned}
&\min_{r} \min_{\mathbf{c}} \sum_{k=1}^{K} \left(\sum_{j=1}^{J} c_j \sum_{i=1}^{n_k} \left[B_j\left(\frac{i}{n_k}\right) - B_j\left(\frac{i-1}{n_k}\right)\right] (z_{ik})^r - y_k^r \right)^2 \\
&\text{s.t.} \qquad \sum_{j=1}^{J} c_j B_j(0) = 0, \quad \sum_{j=1}^{J} c_j B_j(1) = 1, \\
&\qquad\qquad c_j \geq 0,
\end{aligned}$$

in which at the inner level we solve a QP problem with a fixed r, and at the outer level we optimize with respect to a single nonlinear parameter r, in the same way we did in Example 2.73.

For more complicated case of both generating functions g and Q given non-parametrically (as splines) we refer to [20].

2.6 Choquet Integral

2.6.1 Semantics

In this section we present a large family of aggregation functions based on Choquet integrals. The Choquet integral generalizes the Lebesgue integral, and like it, is defined with respect to a measure. Informally, a measure is a function used to measure, in some sense, sets of objects (finite or infinite). For example, the length of an interval on the real line is an example of a measure, applicable to subsets of real numbers. The area or the volume are other examples of simple measures. A broad overview of various measures is given in [71, 201, 252].

We note that measures can be additive (the measure of a set is the sum of the measures of its non-intersecting subsets) or non-additive. Lengths, areas and volumes are examples of additive measures. Lebesgue integration is defined with respect to additive measures. If a measure is non-additive, then the

measure of the total can be larger or smaller than the sum of the measures of its components.

Choquet integration is defined with respect to not necessarily additive monotone measures, called fuzzy measures (see Definition 2.75 below), or *capacities* [58]. In this book we are interested only in *discrete* fuzzy measures, which are defined on finite discrete subsets. This is because our construction of aggregation functions involves a finite set of inputs. In general Choquet integrals (and also various other fuzzy integrals) are defined for measures on general sets, and we refer the reader to extensive literature, e.g., [71, 116, 201, 252].

The main purpose of Choquet integral-based aggregation is to combine the inputs in such a way that not only the importance of individual inputs (as in weighted means), or of their magnitude (as in OWA), are taken into account, but also of their groups (or coalitions). For example, a particular input may not be important by itself, but become very important in the presence of some other inputs. In medical diagnosis, for instance, some symptoms by themselves may not be really important, but may become key factors in the presence of other signs.

A discrete fuzzy measure allows one to assign importances to all possible groups of criteria, and thus offers a much greater flexibility for modeling aggregation. It also turns out that weighted arithmetic means and OWA are special cases of Choquet integrals with respect to additive and symmetric fuzzy measures respectively. Thus we deal with a much broader class of aggregation functions. The uses of Choquet integrals as aggregation functions are documented in [106, 110, 111, 163, 165].

Example 2.74. [106] Consider the problem of evaluating students in a high school with respect to three subjects: mathematics (M), physics (P) and literature (L). Usually this is done by using a weighted arithmetic mean, whose weights are interpreted as importances of different subjects. However, students that are good at mathematics are usually also good at physics and vice versa, as these disciplines present some overlap. Thus evaluation by a weighted arithmetic mean will be either overestimated or underestimated for students good at mathematics and/or physics, depending on the weights.

Consider three students a, b and c whose marks on the scale from 0 to 20 are given by

Student	M	P	L
a	18	16	10
b	10	12	18
c	14	15	15

Suppose that the school is more scientifically oriented, so it weights M and P more than L, with the weights $w_M = w_P > w_L$. If the school wants to favor well equilibrated students, then student c should be considered better than a, who has weakness in L. However, there is no weighting vector $\mathbf{w}$, such that $w_M = w_P > w_L$, and $M_{\mathbf{w}}(c_M, c_P, c_L) > M_{\mathbf{w}}(a_M, a_P, a_L)$.

By aggregating scores using Choquet integral, it is possible (see Example 2.84 below) to construct such a fuzzy measure, that the weights of individual subjects satisfy the requirement $w_M = w_P > w_L$, but the weight attributed to the pair (M,P) is less that the sum $w_M + w_P$, and the well equilibrated student c is favored.

2.6.2 Definitions and properties

Definition 2.75 (Fuzzy measure). *Let* $\mathcal{N} = \{1, 2, \ldots, n\}$. *A discrete fuzzy measure is a set function*[15] $v : 2^{\mathcal{N}} \to [0, 1]$ *which is monotonic (i.e.* $v(\mathcal{A}) \leq v(\mathcal{B})$ *whenever* $\mathcal{A} \subset \mathcal{B}$*) and satisfies* $v(\emptyset) = 0$ *and* $v(\mathcal{N}) = 1$.

Fuzzy measures are interpreted from various points of view, and are used, in particular, to model uncertainty [71, 86, 116, 252]. In the context of aggregation functions, we are interested in another interpretation, the importance of a coalition, which is used in game theory and in multi–criteria decision making. In the Definition 2.75, a subset $\mathcal{A} \subseteq \mathcal{N}$ can be considered as a *coalition*, so that $v(\mathcal{A})$ gives us an idea about the importance or the weight of this coalition. The monotonicity condition implies that adding new elements to a coalition does not decrease its weight.

Example 2.76. The weakest and the strongest fuzzy measures are, respectively,

1. $v(\mathcal{A}) = \begin{cases} 1, & \text{if } \mathcal{A} = \mathcal{N}\ , \\ 0 & \text{otherwise;} \end{cases}$
2. $v(\mathcal{A}) = \begin{cases} 0, & \text{if } \mathcal{A} = \emptyset, \\ 1 & \text{otherwise.} \end{cases}$

Example 2.77. The Dirac measure is given for any $\mathcal{A} \subseteq \mathcal{N}$ by

$$v(\mathcal{A}) = \begin{cases} 1, & \text{if } x_0 \in \mathcal{A}, \\ 0, & \text{if } x_0 \notin \mathcal{A}, \end{cases}$$

where x_0 is a fixed element in $\mathcal{N}$.

Example 2.78. The expression

$$v(\mathcal{A}) = \left(\frac{\mid \mathcal{A} \mid}{n} \right)^2,$$

where $\mid \mathcal{A} \mid$ is the number of elements in $\mathcal{A}$, is a fuzzy measure.

---

[15] A set function is a function whose domain consists of all possible subsets of $\mathcal{N}$. For example, for $n = 3$, a set function is specified by $2^3 = 8$ values at $v(\emptyset)$, $v(\{1\})$, $v(\{2\})$, $v(\{3\})$, $v(\{1, 2\})$, $v(\{1, 3\})$, $v(\{2, 3\})$, $v(\{1, 2, 3\})$.

Definition 2.79 (Möbius transformation). *Let v be a fuzzy measure* [16]*. The Möbius transformation of v is a set function defined for every $\mathcal{A} \subseteq \mathcal{N}$ as*

$$\mathcal{M}(\mathcal{A}) = \sum_{\mathcal{B} \subseteq \mathcal{A}} (-1)^{|\mathcal{A} \setminus \mathcal{B}|} v(\mathcal{B}).$$

Möbius transformation is invertible, and one recovers v by using its inverse, called *Zeta transform*,

$$v(\mathcal{A}) = \sum_{\mathcal{B} \subseteq \mathcal{A}} \mathcal{M}(B) \quad \forall \mathcal{A} \subseteq \mathcal{N}.$$

Möbius transformation is helpful in expressing various quantities, like the interaction indices discussed in Section 2.6.4, in a more compact form [107, 108, 165]. It also serves as an alternative representation of a fuzzy measure, called Möbius representation. That is, one can either use v or $\mathcal{M}$ to perform calculations, whichever is more convenient. The conditions of monotonicity of a fuzzy measure, and the boundary conditions $v(\emptyset) = 0, v(\mathcal{N}) = 1$ are expressed, respectively, as

$$\sum_{\mathcal{B} \subseteq A | i \in \mathcal{B}} \mathcal{M}(\mathcal{B}) \geq 0, \quad \text{for all } \mathcal{A} \subseteq \mathcal{N} \text{ and all } i \in \mathcal{A}, \tag{2.54}$$
$$\mathcal{M}(\emptyset) = 0 \text{ and } \sum_{\mathcal{A} \subseteq \mathcal{N}} \mathcal{M}(\mathcal{A}) = 1.$$

To represent set functions (for a small n), it is convenient to arrange their values into an array [17], e.g., for $n = 3$

$$\begin{array}{ccc} & v(\{1,2,3\}) & \\ v(\{1,2\}) & v(\{1,3\}) & v(\{2,3\}) \\ v(\{1\}) & v(\{2\}) & v(\{3\}) \\ & v(\emptyset) & \end{array}$$

Example 2.80. Let v be the fuzzy measure on $\mathcal{N} = \{1, 2, 3\}$ given by

$$\begin{array}{ccc} & 1 & \\ 0.9 & 0.5 & 0.3 \\ 0.5 & 0 & 0.3 \\ & 0 & \end{array}$$

Its Möbius representation $\mathcal{M}$ is

[16] In general, this definition applies to any set function.

[17] Such an array is based on a Hasse diagram of the inclusion relation defined on the set of subsets of $\mathcal{N}$.

$$\begin{array}{ccc} & 0.1 & \\ 0.4 & -0.3 & 0 \\ 0.5 & 0 & 0.3 \\ & 0 & \end{array}$$

Note 2.81. Observe that, the sum of all values of the Möbius transformation in the above example is equal to 1, in accordance with (2.54). The values of v and $\mathcal{M}$ coincide on singletons.

There are various special classes of fuzzy measures, which we discuss in Section 2.6.3. We now proceed with the definition of the Choquet integral–based aggregation functions.

Definition 2.82 (Discrete Choquet integral). *The discrete Choquet integral with respect to a fuzzy measure v is given by*

$$C_v(\mathbf{x}) = \sum_{i=1}^{n} x_{(i)}[v(\{j|x_j \geq x_{(i)}\}) - v(\{j|x_j \geq x_{(i+1)}\})], \tag{2.55}$$

where $\mathbf{x}_{\nearrow} = (x_{(1)}, x_{(2)}, \ldots, x_{(n)})$ is a non-decreasing permutation of the input $\mathbf{x}$, and $x_{(n+1)} = \infty$ by convention.

Alternative expressions

- By rearranging the terms of the sum, (2.55) can also be written as

$$C_v(\mathbf{x}) = \sum_{i=1}^{n} \left[x_{(i)} - x_{(i-1)}\right] v(H_i). \tag{2.56}$$

  where $x_{(0)} = 0$ by convention, and $H_i = \{(i), \ldots, (n)\}$ is the subset of indices of the $n - i + 1$ largest components of $\mathbf{x}$.
- The discrete Choquet integral is a linear function of the values of the fuzzy measure v. Let us define the following function. For each $\mathcal{A} \subseteq \mathcal{N}$ let

$$g_{\mathcal{A}}(\mathbf{x}) = \max(0, \min_{i \in \mathcal{A}} x_i - \max_{i \in \mathcal{N} \setminus \mathcal{A}} x_i), \tag{2.57}$$

  The maximum and minimum over an empty set are taken as 0. Note that $g_{\mathcal{A}}(\mathbf{x}) = 0$ unless $\mathcal{A}$ is the subset of indices of the k largest components of $\mathbf{x}$, in which case $g_{\mathcal{A}}(\mathbf{x}) = x_{\searrow(k)} - x_{\searrow(k+1)}$. Then it is a matter of simple calculation to show that

$$C_v(\mathbf{x}) = \sum_{\mathcal{A} \subseteq \mathcal{N}} v(\mathcal{A}) g_{\mathcal{A}}(\mathbf{x}). \tag{2.58}$$

- Choquet integral can be expressed with the help of the Möbius transformation as

$$C_v(\mathbf{x}) = \sum_{\mathcal{A}\subseteq\mathcal{N}} \mathcal{M}(\mathcal{A}) \min_{i\in\mathcal{A}} x_i = \sum_{\mathcal{A}\subseteq\mathcal{N}} \mathcal{M}(\mathcal{A}) h_{\mathcal{A}}(\mathbf{x}), \tag{2.59}$$

with $h_{\mathcal{A}}(\mathbf{x}) = \min_{i\in\mathcal{A}} x_i$. By using Definition 2.79 we obtain

$$C_v(\mathbf{x}) = \sum_{\mathcal{A}\subseteq\mathcal{N}} v(\mathcal{A}) \sum_{\mathcal{B}|\mathcal{A}\subseteq\mathcal{B}} (-1)^{|\mathcal{B}\setminus\mathcal{A}|} \min_{i\in\mathcal{B}} x_i. \tag{2.60}$$

By comparing this expression with (2.58) we obtain

$$g_{\mathcal{A}}(\mathbf{x}) = \max(0, \min_{i\in\mathcal{A}} x_i - \max_{i\in\mathcal{N}\setminus\mathcal{A}} x_i) = \sum_{\mathcal{B}|\mathcal{A}\subseteq\mathcal{B}} (-1)^{|\mathcal{B}\setminus\mathcal{A}|} h_{\mathcal{B}}(\mathbf{x}). \tag{2.61}$$

Main properties

- The Choquet integral is a continuous piecewise linear idempotent aggregation function;
- An aggregation function is a Choquet integral if and only if it is homogeneous, shift-invariant and *comonotone additive*, i.e., $C_v(\mathbf{x}+\mathbf{y}) = C_v(\mathbf{x}) + C_v(\mathbf{y})$ for all comonotone [18] $\mathbf{x}, \mathbf{y}$;
- The Choquet integral is uniquely defined by its values at the vertices of the unit cube $[0,1]^n$, i.e., at the points $\mathbf{x}$, whose coordinates $x_i \in \{0,1\}$. Note that there are 2^n such points, the same as the number of values that determine the fuzzy measure v;
- Choquet integral is Lipschitz-continuous, with the Lipschitz constant 1 in any p–norm, which means it is a kernel aggregation function, see Definition 1.62, p. 23;
- The class of Choquet integrals includes weighted means and OWA functions, as well as minimum, maximum and order statistics as special cases (see Section 2.6.5 below);
- A linear convex combination of Choquet integrals with respect to fuzzy measures v_1 and v_2, $\alpha C_{v_1} + (1-\alpha) C_{v_2}, \alpha \in [0,1]$, is also a Choquet integral with respect to $v = \alpha v_1 + (1-\alpha) v_2$;[19]
- A pointwise maximum or minimum of Choquet integrals is not necessarily a Choquet integral (but it is in the bivariate case);
- The class of Choquet integrals is closed under duality;

[18] Two vectors $\mathbf{x}, \mathbf{y} \in \Re^n$ are called comonotone if there exists a common permutation P of $\{1, 2, \ldots, n\}$, such that $x_{P(1)} \leq x_{P(2)} \leq \cdots \leq x_{P(n)}$ and $y_{P(1)} \leq y_{P(2)} \leq \cdots \leq y_{P(n)}$. Equivalently, this condition is frequently expressed as $(x_i - x_j)(y_i - y_j) \geq 0$ for all $i, j \in \{1, \ldots, n\}$.

[19] As a consequence, this property holds for a linear convex combination of any number of fuzzy measures.

- Choquet integrals have neutral and absorbent elements only in the limiting cases of min and max.

Other properties of Choquet integrals depend on the fuzzy measure being used. We discuss them in Section 2.6.3.

Calculation

Calculation of the discrete Choquet integral is performed using Equation (2.56) using the following procedure. Consider the vector of pairs $((x_1, 1), (x_2, 2), \dots, (x_n, n))$, where the second component of each pair is just the index i of x_i. The second component will help keeping track of all permutations.

Calculation of $C_v(\mathbf{x})$.

1. Sort the components of $((x_1, 1), (x_2, 2), \dots, (x_n, n))$ with respect to the first component of each pair in non-decreasing order. We obtain $((x_{(1)}, i_1), (x_{(2)}, i_2), \dots, (x_{(n)}, i_n))$, so that $x_{(j)} = x_{i_j}$ and $x_{(j)} \le x_{(j+1)}$ for all i. Let also $x_{(0)} = 0$.
2. Let $\mathcal{T} = \{1, \dots, n\}$, and $S = 0$.
3. For $j = 1, \dots, n$ do
   a) $S := S + [x_{(j)} - x_{(j-1)}]v(\mathcal{T})$;
   b) $\mathcal{T} := \mathcal{T} \setminus \{i_j\}$
4. Return S.

Example 2.83. Let $n = 3$, values of v be given and $\mathbf{x} = (0.8, 0.1, 0.6)$.

Step 1. We take $((0.8, 1), (0.1, 2), (0.6, 3))$.
Sort this vector of pairs to obtain $((0.1, 2), (0.6, 3), (0.8, 1))$.
Step 2. Take $\mathcal{T} = \{1, 2, 3\}$ and $S = 0$.
Step 3. a) $S := 0 + [0.1 - 0]v(\{1, 2, 3\}) = 0.1 \times 1 = 0.1$;
b) $\mathcal{T} = \{1, 2, 3\} \setminus \{2\} = \{1, 3\}$;

a) $S := 0.1 + [0.6 - 0.1]v(\{1, 3\}) = 0.1 + 0.5v(\{1, 3\})$;
b)$\mathcal{T} := \{1, 3\} \setminus \{3\} = \{1\}$;

a) $S := [0.1 + 0.5v(\{1, 3\})] + [0.8 - 0.6]v(\{1\})$.

Therefore $C_v(\mathbf{x}) = 0.1 + 0.5v(\{1, 3\}) + 0.2v(\{1\})$.

For computational purposes it is convenient to store the values of a fuzzy measure v in an array $\mathbf{v}$ of size 2^n, and to use the following indexing system, which provides a one-to-one mapping between the subsets $\mathcal{J} \subseteq \mathcal{N}$ and the set of integers $I = \{0, \dots, 2^n - 1\}$, which index the elements of v. Take the binary representation of each index in I, e.g. $j = 5 = 101$ (binary). Now for a given subset $\mathcal{J} \subseteq \mathcal{N} = \{1, \dots, n\}$ define its characteristic vector $\mathbf{c} \in \{0, 1\}^n$: $c_{n-i+1} = 1$ if $i \in \mathcal{J}$ and 0 otherwise. For example, if $n = 5$, $\mathcal{J} = \{1, 3\}$, then $\mathbf{c} = (0, 0, 1, 0, 1)$. Put the value $v(\mathcal{J})$ into correspondence with v_j, so that the

binary representation of j corresponds to the characteristic vector of $\mathcal{J}$. In our example $v(\{1,3\}) = v_5$.

Such an ordering of the subsets of $\mathcal{N}$ is called binary ordering:

$$\emptyset, \{1\}, \{2\}, \{1,2\}, \{3\}, \{1,3\}, \{2,3\}, \{1,2,3\}, \{4\}, \ldots, \{1,2,\ldots,n\}.$$

The values of v are mapped to the elements of vector $\mathbf{v}$ as follows

v_0 $= v_{(0000)}$	v_1 $= v_{(0001)}$	v_2 $= v_{(0010)}$	v_3 $= v_{(0011)}$	v_4 $= v_{(0100)}$	v_5 $= v_{(0101)}$	$\ldots$
$v(\emptyset)$	$v(\{1\})$	$v(\{2\})$	$v(\{1,2\})$	$v(\{3\})$	$v(\{1,3\})$	$\ldots$

Using (2.58) and the above indexing system, we can write

$$C_v(\mathbf{x}) = \sum_{j=0}^{2^n-1} v_j g_j(\mathbf{x}) = < \mathbf{g}(\mathbf{x}), \mathbf{v} >, \tag{2.62}$$

where as earlier, functions $g_j, j = 0, \ldots, 2^n - 1$ are defined by

$$g_j(\mathbf{x}) = \max(0, \min_{i \in \mathcal{J}} x_i - \max_{i \in \mathcal{N} \setminus \mathcal{J}} x_i), \tag{2.63}$$

and the characteristic vector of the set $\mathcal{J} \subseteq \mathcal{N}$ corresponds to the binary representation of j.

An alternative ordering of the values of v is based on set cardinality:

$$\emptyset, \underbrace{\{1\}, \{2\}, \ldots, \{n\}}_{n \text{ singletons}}, \underbrace{\{1,2\}, \{1,3\}, \ldots, \{1,n\}, \{2,3\}, \ldots, \{n-1,n\}}_{\binom{n}{2} \text{ pairs}}, \{1,2,3\}, \ldots.$$

Such an ordering is useful when dealing with k-additive fuzzy measures (see Definition 2.121 and Proposition 2.134 below), as it allows one to group non-zero values $\mathcal{M}(\mathcal{A})$ (in Möbius representation) at the beginning of the array. We shall discuss these orderings in Section 2.6.6.

Example 2.84. [106] We continue Example 2.74 on p. 91. Let the fuzzy measure v be given as

$$\begin{array}{ccc} & 1 & \\ 0.5 & 0.9 & 0.9 \\ 0.45 & 0.45 & 0.3 \\ & 0 & \end{array}$$

so that the ratio of weights $w_M : w_P : w_L = 3 : 3 : 2$, but since mathematics and physics overlap, the weight of the pair $v(\{M,P\}) = 0.5 < v(\{M\}) + v(\{P\})$. On the other hand, weights attributed to $v(\{M,L\})$ and $v(\{P,L\})$ are greater than the sum of individual weights.

Using Choquet integral, we obtain the global scores [20] $C_v(a_M, a_P, a_L) = 13.9$, $C_v(b_M, b_P, b_L) = 13.6$ and $C_v(c_M, c_P, c_L) = 14.6$, so that the students

[20] Since Choquet integral is a homogeneous aggregation function, we can calculate it directly on $[0,20]^n$ rather than scaling the inputs to $[0,1]^n$.

are ranked as $b \prec a \prec c$ as required. Student b has the lowest rank as requested by the scientific tendency of the school.

2.6.3 Types of fuzzy measures

The properties of the Choquet integral depend on the fuzzy measure v being used. There are various generic types of fuzzy measures, which lead to specific features of Choquet integral-based aggregation, and to several special cases, such as weighted arithmetic means, OWA and WOWA discussed earlier in this Chapter (see also Section 2.6.5). In this section we present the most important definitions and classes of fuzzy measures.

Definition 2.85 (Dual fuzzy measure). *Given a fuzzy measure v, its dual fuzzy measure v^* is defined by*

$$v^*(\mathcal{A}) = 1 - v(\mathcal{A}^c), \text{ for all } \mathcal{A} \subseteq \mathcal{N},$$

where $\mathcal{A}^c = \mathcal{N} \setminus \mathcal{A}$ is the complement of $\mathcal{A}$ in $\mathcal{N}$.

Definition 2.86 (Self–dual fuzzy measure). *A fuzzy measure v is self-dual if it is equal to its dual $v*$, i.e.,*

$$v(\mathcal{A}) + v(\mathcal{A}^c) = 1, \text{holds for all } A \subseteq \mathcal{N} \ .$$

Definition 2.87 (Submodular and supermodular fuzzy measure). *A fuzzy measure v is called submodular if for any $\mathcal{A}, \mathcal{B} \subseteq \mathcal{N}$*

$$v(\mathcal{A} \cup \mathcal{B}) + v(\mathcal{A} \cap \mathcal{B}) \leq v(\mathcal{A}) + v(\mathcal{B}). \tag{2.64}$$

It is called supermodular if

$$v(\mathcal{A} \cup \mathcal{B}) + v(\mathcal{A} \cap \mathcal{B}) \geq v(\mathcal{A}) + v(\mathcal{B}). \tag{2.65}$$

Two weaker conditions which are frequently used are called sub- and super-additivity. These are special cases of sub- and supermodularity for disjoint subsets

---

Definition 2.88 (Subadditive and superadditive fuzzy measure). *A fuzzy measure v is called subadditive if for any two nonintersecting subsets $\mathcal{A}, \mathcal{B} \subset \mathcal{N}$, $\mathcal{A} \cap \mathcal{B} = \emptyset$:*

$$v(\mathcal{A} \cup \mathcal{B}) \leq v(\mathcal{A}) + v(\mathcal{B}). \tag{2.66}$$

It is called superadditive if

$$v(\mathcal{A} \cup \mathcal{B}) \geq v(\mathcal{A}) + v(\mathcal{B}). \tag{2.67}$$

Note 2.89. Clearly sub-(super-) modularity implies sub-(super-) additivity.

Note 2.90. A fuzzy measure is supermodular if and only if its dual is submodular. However the dual of a subadditive fuzzy measure is not necessarily superadditive and vice versa.

Note 2.91. A general fuzzy measure may be submodular only with respect to specific pairs of subsets $\mathcal{A}, \mathcal{B}$, and supermodular with respect to other pairs.

Definition 2.92 (Additive (probability) measure). *A fuzzy measure v is called **additive** if for any $\mathcal{A}, \mathcal{B} \subset \mathcal{N}$, $\mathcal{A} \cap \mathcal{B} = \emptyset$:*

$$v(\mathcal{A} \cup \mathcal{B}) = v(\mathcal{A}) + v(\mathcal{B}). \tag{2.68}$$

An additive fuzzy measure is called a **probability** *measure.*

Note 2.93. A fuzzy measure is both sub- and supermodular if and only if it is additive. A fuzzy measure is both sub- and superadditive if and only if it is additive.

Note 2.94. For an additive fuzzy measure clearly $v(\mathcal{A}) = \sum_{i \in \mathcal{A}} v(\{i\})$.

Note 2.95. Additivity implies that for any subset $\mathcal{A} \subseteq \mathcal{N} \setminus \{i, j\}$

$$v(\mathcal{A} \cup \{i, j\}) = v(\mathcal{A} \cup \{i\}) + v(\mathcal{A} \cup \{j\}) - v(\mathcal{A}).$$

Definition 2.96 (Boolean measure). *A fuzzy measure v is called a boolean fuzzy measure or $\{0,1\}$–measure if it holds:*

$$v(\mathcal{A}) = 0 \text{ or } v(\mathcal{A}) = 1,$$

for all $\mathcal{A} \subseteq \mathcal{N}$.

Definition 2.97 (Balanced measure). *A fuzzy measure v is called balanced if it holds:*

$$\mid \mathcal{A} \mid < \mid \mathcal{B} \mid \Longrightarrow v(\mathcal{A}) \le v(\mathcal{B}), \text{ for all } \mathcal{A}, \mathcal{B} \subseteq \mathcal{N}.$$

Definition 2.98 (Symmetric fuzzy measure). *A fuzzy measure v is called symmetric if the value $v(\mathcal{A})$ depends only on the cardinality of the set $\mathcal{A}$, i.e., for any $\mathcal{A}, \mathcal{B} \subseteq \mathcal{N}$,*

$$\text{if } |\mathcal{A}| = |\mathcal{B}| \text{ then } v(\mathcal{A}) = v(\mathcal{B}).$$

Alternatively, one can say that a fuzzy measure v is symmetric if for any $\mathcal{A} \subseteq \mathcal{N}$ it is

$$v(\mathcal{A}) = Q\left(\frac{|\mathcal{A}|}{n}\right), \tag{2.69}$$

for some monotone non-decreasing function $Q : [0,1] \to [0,1], Q(0) = 0$ and $Q(1) = 1$.

Example 2.99. The following fuzzy measure is additive

$$\begin{array}{ccc} & 1 & \\ 0.4 & 0.7 & 0.9 \\ 0.1 & 0.3 & 0.6 \\ & 0 & \end{array}$$

The following fuzzy measure is symmetric

$$\begin{array}{ccc} & 1 & \\ 0.7 & 0.7 & 0.7 \\ 0.2 & 0.2 & 0.2 \\ & 0 & \end{array}$$

The following fuzzy measure is superadditive but not submodular

$$\begin{array}{ccc} & 1 & \\ 0.6 & 0.5 & 0.6 \\ 0.3 & 0.1 & 0.2 \\ & 0 & \end{array}$$

The following fuzzy measure is subadditive and symmetric

$$\begin{array}{ccc} & 1 & \\ 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 \\ & 0 & \end{array}$$

Example 2.100. Let v be $\{0,1\}$–fuzzy measure on $\mathcal{N} = \{1,2,3\}$

$$\begin{array}{ccc} & 1 & \\ 1 & 1 & 0 \\ 0 & 0 & 0 \\ & 0 & \end{array}$$

This measure is superadditive but its dual fuzzy measure v^*, given by

$$\begin{array}{ccc} & 1 & \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ & 0 & \end{array}$$

is not subadditive, because, for instance, $v^*(\{2,3\}) = 1$ and $v^*(\{2\}) + v^*(\{3\}) = 0$, nor is it superadditive, because $v^*(\{1,2,3\}) < v^*(\{1\}) + v^*(\{2,3\})$.

Definition 2.101 (Possibility and necessity measures). *A fuzzy measure is called a possibility, Pos, if for all $\mathcal{A}, \mathcal{B} \subseteq \mathcal{N}$ it satisfies*

$$Pos(\mathcal{A} \cup \mathcal{B}) = \max\{Pos(\mathcal{A}), Pos(\mathcal{B})\}.$$

A fuzzy measure is called a necessity, Nec, if for all $\mathcal{A}, \mathcal{B} \subseteq \mathcal{N}$ it satisfies

$$Nec(\mathcal{A} \cap \mathcal{B}) = \min\{Nec(\mathcal{A}), Nec(\mathcal{B})\}.$$

Note 2.102. Possibility and necessity measures are dual to each other in the sense of Definition 2.85, that is, for all $\mathcal{A} \subseteq \mathcal{N}$

$$Nec(\mathcal{A}) = 1 - Pos(\mathcal{A}^c).$$

A possibility measure is subadditive. A necessity measure is superadditive.

Possibility and necessity measures are the basis of the theory of possibility [85, 252, 284].

Example 2.103. The following fuzzy measure v is a possibility measure

$$\begin{array}{ccc} & 1 & \\ 1 & 0.3 & 1 \\ 0.3 & 1 & 0.2 \\ & 0 & \end{array}$$

Example 2.104. The following fuzzy measure v is a necessity measure, dual to the possibility measure in the previous example

$$\begin{array}{ccc} & 1 & \\ 0.8 & 0 & 0.7 \\ 0 & 0.7 & 0 \\ & 0 & \end{array}$$

Definition 2.105 (Belief Measure). *A belief measure $Bel : 2^{\mathcal{N}} \to [0,1]$ is a fuzzy measure that satisfies the following condition: for all $m > 1$*

$$Bel(\bigcup_{i=1}^{m} \mathcal{A}_i) \geq \sum_{\emptyset \neq I \subset \{1,\ldots,m\}} (-1)^{|I|+1} Bel(\bigcap_{i \in I} \mathcal{A}_i),$$

where $\{\mathcal{A}_i\}_{i \in \{1,\ldots,m\}}$, is any finite family of subsets of $\mathcal{N}$. [21]

[21] For a fixed $m \geq 1$ this condition is called m-monotonicity (simple monotonicity for $m = 1$), and if it holds for all $m \geq 1$, it is called total monotonicity [72, 108]. For a fixed m, condition in Definition 2.106 is called m-alternating monotonicity. 2-monotone fuzzy measures are called supermodular (see Definition 2.87), also called convex, whereas 2-alternating fuzzy measures are called submodular. If a fuzzy measure is m-monotone, its dual is m-alternating and vice versa.

Definition 2.106 (Plausibility measure). *A plausibility measure $Pl : 2^{\mathcal{N}} \to [0,1]$ is a fuzzy measure that satisfies the following condition: for all $m > 1$*

$$Pl(\bigcap_{i=1}^{m} \mathcal{A}_i) \leq \sum_{\emptyset \neq I \subset \{1,\ldots,m\}} (-1)^{|I|+1} Pl(\bigcup_{i \in I} \mathcal{A}_i),$$

where $\{\mathcal{A}_i\}_{i \in \{1,\ldots,m\}}$ is any finite family of subsets of $\mathcal{N}$.

Belief and *plausibility* measures constitute the basis of Dempster and Shafer Evidence Theory [222]. Belief measures are related to (and sometimes defined through) *basic probability assignments*, which are the values of the Möbius transformation. We refer the reader to the literature in this field, e.g., [116, 252].

Note 2.107. A set function $Pl : 2^{\mathcal{N}} \to [0,1]$ is a plausibility measure if its dual set function is a belief measure, i.e., for all $\mathcal{A} \subseteq \mathcal{N}$

$$Pl(\mathcal{A}) = 1 - Bel(\mathcal{A}^c).$$

Any belief measure is superadditive. Any plausibility measure is subadditive.

Note 2.108. A fuzzy measure is both a belief and a plausibility measure if and only if it is additive.

Note 2.109. A possibility measure is a plausibility measure and a necessity measure is a belief measure.

Note 2.110. The set of all fuzzy measures (for a fixed $\mathcal{N}$) is convex[22]. The sets of subadditive, superadditive, submodular, supermodular, subadditive, superadditive, additive, belief and plausibility fuzzy measures are convex. However the sets of possibility and necessity measures are not convex.

λ-fuzzy measures

Additive and symmetric fuzzy measures are two examples of very simple fuzzy measures, whereas general fuzzy measures are sometimes too complicated for applications. Next we examine some fuzzy measures with intermediate complexity, which are powerful enough to express interactions among the variables, yet require much less than 2^n parameters to express them.

As a way of reducing the complexity of a fuzzy measure Sugeno [230] introduced λ-fuzzy measures (also called Sugeno measures).

Definition 2.111 (λ-fuzzy measure). *Given a parameter $\lambda \in]-1, \infty[$, a λ-fuzzy measure is a fuzzy measure v that for all $\mathcal{A}, \mathcal{B} \subseteq \mathcal{N}, \mathcal{A} \cap \mathcal{B} = \emptyset$ satisfies*

$$v(\mathcal{A} \cup \mathcal{B}) = v(\mathcal{A}) + v(\mathcal{B}) + \lambda v(\mathcal{A}) v(\mathcal{B}). \tag{2.70}$$

[22] A set E is convex if $\alpha x + (1-\alpha) y \in E$ for all $x, y \in E, \alpha \in [0,1]$.

Under these conditions, all the values $v(\mathcal{A})$ are immediately computed from n independent values $v(\{i\}), i = 1, \ldots, n$, by using the explicit formula

$$v(\bigcup_{i=1}^{m}\{i\}) = \frac{1}{\lambda}\left(\prod_{i=1}^{m}(1 + \lambda v(\{i\})) - 1\right), \quad \lambda \neq 0.$$

If $\lambda = 0$, λ-fuzzy measure becomes a probability measure. The coefficient λ is determined from the boundary condition $v(\mathcal{N}) = 1$, which gives

$$\lambda + 1 = \prod_{i=1}^{n}(1 + \lambda v(\{i\})), \tag{2.71}$$

which can be solved on $(-1, 0)$ or $(0, \infty)$ numerically (note that $\lambda = 0$ is always a solution). Thus a λ-fuzzy measure is characterized by n independent values $v(\{i\}), i = 1, \ldots, n$.

A λ-fuzzy measure v is related to a probability measure P through the relation

$$P(\mathcal{A}) = \frac{\log(1 + \lambda v(\mathcal{A}))}{1 + \lambda},$$

and, using $g(t) = ((1 + \lambda)^t - 1)/\lambda$ for $\lambda > -1, \lambda \neq 0$, and $g(t) = t$ for $\lambda = 0$,

$$g(P(\mathcal{A})) = v(\mathcal{A}).$$

Note 2.112. The set of all λ-fuzzy measures is not convex.

A λ-fuzzy measure is an example of a distorted probability measure.

Definition 2.113 (Distorted probability measure). *A fuzzy measure v is a distorted probability measure if there exists some non-decreasing function $g : [0, 1] \to [0, 1]$, $g(0) = 0, g(1) = 1$, and a probability measure P, such that for all $\mathcal{A} \subset \mathcal{N}$:*

$$v(\mathcal{A}) = g(P(\mathcal{A})).$$

We remind that Weighted OWA functions (see p. 72) are equivalent to Choquet integrals with respect to distorted probabilities. Distorted probabilities and their extension, m-dimensional distorted probabilities, have been recently studied in [193].

A λ-fuzzy measure is also an example of a decomposable fuzzy measure [117].

Definition 2.114 (Decomposable fuzzy measure). *A decomposable fuzzy measure v is a fuzzy measure which for all $\mathcal{A}, \mathcal{B} \subseteq \mathcal{N}, \mathcal{A} \cap \mathcal{B} = \emptyset$ satisfies*

$$v(\mathcal{A} \cup \mathcal{B}) = f(v(\mathcal{A}), v(\mathcal{B})) \tag{2.72}$$

for some function $f : [0, 1]^2 \to [0, 1]$ known as the decomposition function.

Note 2.115. It turns out that to get $v(\mathcal{N}) = 1$, f must necessarily be a t–conorm (see Chapter 3). In the case of λ-fuzzy measures, f is an Archimedean t–conorm with an additive generator $h(t) = \frac{\ln(1+\lambda t)}{\ln(1+\lambda)}$, $\lambda \neq 0$, which is a Sugeno-Weber t–conorm, see p. 162.

Note 2.116. Additive measures are decomposable with respect to the Łukasiewicz t–conorm $S_L(x,y) = \min(1, x+y)$. But not every S_L-decomposable fuzzy measure is a probability. Possibility measures are decomposable with respect to the maximum t–conorm $S_{max}(x,y) = \max(x,y)$. Every $S_{\max}$-decomposable discrete fuzzy measure is a possibility measure.

Note 2.117. A λ-fuzzy measure is either sub- or supermodular, when $-1 < \lambda \leq 0$ or $\lambda \geq 0$ respectively.

Note 2.118. When $-1 < \lambda \leq 0$, a λ-fuzzy measure is a plausibility measure, and when $\lambda \geq 0$ it is a belief measure.

Note 2.119. Dirac measures (Example 2.77) are λ-fuzzy measures for all $\lambda \in]-1, \infty[$.

Note 2.120. For a given t–conorm S and fixed $\mathcal{N}$, the set of all S-decomposable fuzzy measures is not always convex.

k - additive fuzzy measures

Another way to reduce complexity of aggregation functions based on fuzzy measures is to impose various linear constraints on their values. Such constraints acquire an interesting interpretation in terms of interaction indices discussed in the next section. One type of constraints leads to k-additive fuzzy measures.

Definition 2.121 (k-additive fuzzy measure). *A fuzzy measure v is called k-additive ($1 \leq k \leq n$) if its Möbius transformation verifies*

$$\mathcal{M}(\mathcal{A}) = 0$$

for any subset $\mathcal{A}$ with more than k elements, $|\mathcal{A}| > k$, and there exists a subset $\mathcal{B}$ with k elements such that $\mathcal{M}(\mathcal{B}) \neq 0$.

An alternative definition of k-additivity (which is also applicable to fuzzy measures on more general sets than $\mathcal{N}$) was given by Mesiar in [181, 182]. It involves a weakly monotone[23] additive set function v_k defined on subsets of $\mathcal{N}^k = \mathcal{N} \times \mathcal{N} \times \ldots \times \mathcal{N}$. A fuzzy measure v is k-additive if $v(\mathcal{A}) = v_k(\mathcal{A}^k)$ for all $\mathcal{A} \subseteq \mathcal{N}$.

[23] Weakly monotone means $\forall \mathcal{A}, \mathcal{B} \subseteq \mathcal{N}$, $\mathcal{A} \subseteq \mathcal{B}$ implies $v_k(\mathcal{A}^k) \leq v_k(\mathcal{B}^k)$.

2.6.4 Interaction, importance and other indices

When dealing with multiple criteria, it is often the case that these are not independent, and there is some interaction (positive or negative) among the criteria. For instance, two or more criteria may point essentially to the same concept, for example criteria such as "learnability" and "memorability" that are used to evaluate software user interface [226]. If the criteria are combined by using, e.g., weighted means, their scores will be double counted. In other instances, contribution of one criterion to the total score by itself may be small, but sharply rise when taken in conjunction with other criteria (i.e., in a "coalition")[24].

Thus to measure such concepts as the importance of a criterion and interaction among the criteria, we need to account for contribution of these criteria in various coalitions. To do this we will use the concepts of Shapley value, which measures the importance of a criterion i in all possible coalitions, and the interaction index, which measures the interaction of a pair of criteria i, j in all possible coalitions [107, 108].

Definition 2.122 (Shapley value). *Let v be a fuzzy measure. The Shapley index for every $i \in \mathcal{N}$ is*

$$\phi(i) = \sum_{\mathcal{A} \subseteq \mathcal{N} \setminus \{i\}} \frac{(n - |\mathcal{A}| - 1)!|\mathcal{A}|!}{n!} [v(\mathcal{A} \cup \{i\}) - v(\mathcal{A})].$$

The Shapley value is the vector $\boldsymbol{\phi}(v) = (\phi(1), \ldots, \phi(n))$.

Note 2.123. It is informative to write the Shapley index as

$$\phi(i) = \frac{1}{n} \sum_{t=0}^{n-1} \frac{1}{\binom{n-1}{t}} \sum_{\mathcal{A} \subseteq \mathcal{N} \setminus \{i\}, |\mathcal{A}| = t} [v(\mathcal{A} \cup \{i\}) - v(\mathcal{A})].$$

Note 2.124. For an *additive* fuzzy measure we have $\phi(i) = v(\{i\})$.

The Shapley value is interpreted as a kind of average value of the contribution of each criterion alone in all coalitions.

Definition 2.125 (Interaction index). *Let v be a fuzzy measure. The interaction index for every pair $i, j \in \mathcal{N}$ is*

$$I_{ij} = \sum_{\mathcal{A} \subseteq \mathcal{N} \setminus \{i,j\}} \frac{(n - |\mathcal{A}| - 2)!|\mathcal{A}|!}{(n-1)!} [v(\mathcal{A} \cup \{i,j\}) - v(\mathcal{A} \cup \{i\}) - v(\mathcal{A} \cup \{j\}) + v(\mathcal{A})].$$

[24] Such interactions are well known in game theory. For example, contributions of the efforts of workers in a group can be greater or smaller than the sum of their separate contributions (if working independently).

The interaction indices verify $I_{ij} < 0$ as soon as i, j are positively correlated (negative synergy, redundancy). Similarly $I_{ij} > 0$ for negatively correlated criteria (positive synergy, complementarity). $I_{ij} \in [-1, 1]$ for any pair i, j.

Proposition 2.126. *For a submodular fuzzy measure v, all interaction indices verify $I_{ij} \leq 0$. For a supermodular fuzzy measure, all interaction indices verify $I_{ij} \geq 0$.*

The Definition 2.125 due to Murofushi and Soneda was extended by Grabisch for any coalition $\mathcal{A}$ of the criteria (not just pairs) [107].

Definition 2.127 (Interaction index for coalitions). *Let v be a fuzzy measure. The interaction index for every set $\mathcal{A} \subseteq \mathcal{N}$ is*

$$I(\mathcal{A}) = \sum_{\mathcal{B} \subseteq \mathcal{N} \setminus \mathcal{A}} \frac{(n - |\mathcal{B}| - |\mathcal{A}|)!|\mathcal{B}|!}{(n - |\mathcal{A}| + 1)!} \sum_{\mathcal{C} \subseteq \mathcal{A}} (-1)^{|\mathcal{A} \setminus \mathcal{C}|} v(\mathcal{B} \cup \mathcal{C}).$$

Note 2.128. Clearly $I(\mathcal{A})$ coincides with I_{ij} if $\mathcal{A} = \{i, j\}$, and coincides with $\phi(i)$ if $\mathcal{A} = \{i\}$. Also $I(\mathcal{A})$ satisfies the dummy criterion axiom: If i is a dummy criterion, i.e., $v(\mathcal{B} \cup \{i\}) = v(\mathcal{B}) + v(\{i\})$ for any $\mathcal{B} \subset \mathcal{N} \setminus \{i\}$, then for every such $\mathcal{B} \neq \emptyset$, $I(\mathcal{B} \cup \{i\}) = 0$. A dummy criterion does not interact with other criteria in any coalition.

Note 2.129. An alternative single-sum expression for $I(\mathcal{A})$ was obtained in [115]:

$$I(\mathcal{A}) = \sum_{\mathcal{B} \subseteq \mathcal{N}} \frac{(-1)^{|\mathcal{A} \setminus \mathcal{B}|}}{(n - |\mathcal{A}| + 1)\binom{n-|\mathcal{A}|}{|\mathcal{B} \setminus \mathcal{A}|}} v(\mathcal{B}).$$

An alternative to the Shapley value is the Banzhaf index [10]. It measures the same concept as the Shapley index, but weights the terms $[v(\mathcal{A} \cup \{i\}) - v(\mathcal{A})]$ in the sum equally.

Definition 2.130 (Banzhaf Index). *Let v be a fuzzy measure. The Banzhaf index b_i for every $i \in \mathcal{N}$ is*

$$b_i = \frac{1}{2^{n-1}} \sum_{\mathcal{A} \subseteq \mathcal{N} \setminus \{i\}} [v(\mathcal{A} \cup \{i\}) - v(\mathcal{A})].$$

This definition has been generalized by Roubens in [211].

Definition 2.131 (Banzhaf interaction index for coalitions). *Let v be a fuzzy measure. The Banzhaf interaction index between the elements of $\mathcal{A} \subseteq \mathcal{N}$ is given by*

$$J(\mathcal{A}) = \frac{1}{2^{n-|\mathcal{A}|}} \sum_{\mathcal{B} \subseteq \mathcal{N} \setminus \mathcal{A}} \sum_{\mathcal{C} \subseteq \mathcal{A}} (-1)^{|\mathcal{A} \setminus \mathcal{C}|} v(\mathcal{B} \cup \mathcal{C}).$$

Note 2.132. An alternative single-sum expression for $J(\mathcal{A})$ was obtained in [115]:

$$J(\mathcal{A}) = \frac{1}{2^{n-|\mathcal{A}|}} \sum_{\mathcal{B} \subseteq \mathcal{N}} (-1)^{|\mathcal{A} \setminus \mathcal{B}|} v(\mathcal{B}).$$

Möbius transformation help one to express the indices mentioned above in a more compact form [107, 108, 115, 165], namely

$$\phi(i) = \sum_{\mathcal{B}|\ i \in \mathcal{B}} \frac{1}{|\mathcal{B}|} \mathcal{M}(\mathcal{B}),$$

$$I(\mathcal{A}) = \sum_{\mathcal{B}|\mathcal{A} \subseteq \mathcal{B}} \frac{1}{|\mathcal{B}| - |\mathcal{A}| + 1} \mathcal{M}(\mathcal{B}),$$

$$J(\mathcal{A}) = \sum_{\mathcal{B}|\mathcal{A} \subseteq \mathcal{B}} \frac{1}{2^{|\mathcal{B}| - |\mathcal{A}|}} \mathcal{M}(\mathcal{B}).$$

Example 2.133. Let v be the fuzzy measure defined as follows:

$$\begin{array}{ccc} & 1 & \\ 0.5 & 0.6 & 0.7 \\ 0.1 & 0.2 & 0.3 \\ & 0 & \end{array}$$

Then the Shapley indices are $\phi(1) = 0.7/3,\ \phi(2) = 1/3,\ \phi(3) = 1.3/3$.

The next result due to Grabisch [107, 108] establishes a fundamental property of k-additive fuzzy measures, which justifies their use in simplifying interactions between the criteria in multiple criteria decision making.

Proposition 2.134. *Let v be a k-additive fuzzy measure, $1 \le k \le n$. Then*

- $I(\mathcal{A}) = 0$ *for every* $\mathcal{A} \subseteq \mathcal{N}$ *such that* $|\mathcal{A}| > k$;
- $I(\mathcal{A}) = J(\mathcal{A}) = \mathcal{M}(\mathcal{A})$ *for every* $\mathcal{A} \subseteq \mathcal{N}$ *such that* $|\mathcal{A}| = k$.

Thus k-additive measures acquire an interesting interpretation. These are fuzzy measures that limit interaction among the criteria to groups of size at most k. For instance, for 2-additive fuzzy measures, there are pairwise interactions among the criteria but no interactions in groups of 3 or more. By limiting the class of fuzzy measures to k-additive measures, one reduces their complexity (the number of values) by imposing linear equality constraints. The total number of linearly independent values is reduced from $2^n - 1$ to $\sum_{i=1}^{k} \binom{n}{i} - 1$.

Orness value

We recall the definition of the measure of orness of an aggregation function on p. 40. By using the Möbius transform one can calculate the orness of a Choquet integral C_v with respect to a fuzzy measure v as follows.

Proposition 2.135 (Orness of Choquet integral). *[167] For any fuzzy measure v the orness of the Choquet integral with respect to v is*

$$orness(C_v) = \frac{1}{n-1} \sum_{\mathcal{A} \subseteq \mathcal{N}} \frac{n - |\mathcal{A}|}{|\mathcal{A}| + 1} \mathcal{M}(\mathcal{A}),$$

where $\mathcal{M}(\mathcal{A})$ is the Möbius representation of $\mathcal{A}$. In terms of v the orness value is

$$orness(C_v) = \frac{1}{n-1} \sum_{\mathcal{A} \subseteq \mathcal{N}} \frac{(n - |\mathcal{A}|)!|\mathcal{A}|!}{n!} v(\mathcal{A}).$$

Another (simplified) criterion which measures positive interaction among pairs of criteria is based on the degree of substitutivity [25].

Definition 2.136 (Substitutive criteria). *Let i, j be two criteria, and let $\nu_{ij} \in [0, 1]$ be the degrees of substitutivity. The fuzzy measure v is called substitutive with respect to the criteria i, j if for any subset $\mathcal{A} \subseteq \mathcal{N} \setminus \{i, j\}$*

$$\begin{aligned} v(\mathcal{A} \cup \{i,j\}) &\le v(\mathcal{A} \cup \{i\}) + (1 - \nu_{ij}) v(\mathcal{A} \cup \{j\}), \\ v(\mathcal{A} \cup \{i,j\}) &\le v(\mathcal{A} \cup \{j\}) + (1 - \nu_{ij}) v(\mathcal{A} \cup \{i\}). \end{aligned} \tag{2.73}$$

Note 2.137. When $\nu_{ij} = 1$, and in view of the monotonicity of fuzzy measures, we obtain the equalities $v(\mathcal{A} \cup \{i,j\}) = v(\mathcal{A} \cup \{i\}) = v(\mathcal{A} \cup \{j\})$, i.e., fully substitutive (identical) criteria. One of these criteria can be seen as dummy.

Note 2.138. If $\nu_{ij} = 0$, i.e., the criteria are not positively substitutive, then $v(\mathcal{A} \cup \{i,j\}) \le v(\mathcal{A} \cup \{i\}) + v(\mathcal{A} \cup \{j\})$, which is a weaker version of (2.66). It does not imply independence, as the criteria may have negative interaction. Also note that it does not imply subadditivity, as $v(\mathcal{A}) \ge 0$, and it only applies to one particular pair of criteria, i, j, not to all pairs. On the other hand subadditivity implies (2.73) for all pairs of criteria, and with some $\nu_{ij} > 0$, i.e., all the criteria are substitutive to some degree.

Entropy

The issue of the entropy of Choquet integrals was treated in [147, 166].

---

[25] See discussion in [117], p.318.

Definition 2.139. *The entropy of a fuzzy measure v is*

$$H(v) = \sum_{i \in \mathcal{N}} \sum_{\mathcal{A} \subseteq \mathcal{N} \setminus \{i\}} \frac{(n - |\mathcal{A}| - 1)!|\mathcal{A}|!}{n!} h(v(\mathcal{A} \cup \{i\}) - v(\mathcal{A})),$$

with $h(t) = -t \log t$, if $t > 0$ and $h(0) = 0$.

This definition coincides with the definition of weights dispersion (Definition 2.37) used for weighted arithmetic mean and OWA functions, when v is additive or symmetric (see Proposition 2.143 below). The maximal value of H is $\log n$ and is achieved if and only if v is an additive symmetric fuzzy measure , i.e., $v(\mathcal{A}) = \frac{|\mathcal{A}|}{n}$ for all $\mathcal{A} \subseteq \mathcal{N}$. The minimal value 0 is achieved if and only if v is a Boolean fuzzy measure. Also H is a strictly concave function of v, which is useful when maximizing H over a convex subset of fuzzy measures, as it leads to a unique global maximum.

2.6.5 Special cases of the Choquet integral

Let us now study special cases of Choquet integral with respect to fuzzy measures with specific properties.

Proposition 2.140. *If v^* is a fuzzy measure dual to a fuzzy measure v, the Choquet integrals C_v and C_{v^*} are dual to each other. If v is self-dual, then C_v is a self-dual aggregation function.*

Proposition 2.141. *The Choquet integral with respect to an additive fuzzy measure v is the weighted arithmetic mean $M_{\mathbf{w}}$ with the weights $w_i = v(\{i\})$.*

Proposition 2.142. *The Choquet integral with respect to a symmetric fuzzy measure v defined by means of a quantifier Q as in (2.69) is the OWA function $OWA_{\mathbf{w}}$ with the weights* [26] $w_i = Q(\frac{i}{n}) - Q(\frac{i-1}{n})$.

The values of the fuzzy measure v, associated with an OWA function with the weighting vector $\mathbf{w}$, are also expressed as

$$v(\mathcal{A}) = \sum_{i=n-|\mathcal{A}|+1}^{n} w_i.$$

If a fuzzy measure is symmetric and additive at the same time, we have

[26] We remind that in the definition OWA, we used a non-increasing permutation of the components of $\mathbf{x}$, $\mathbf{x}_\searrow$, whereas in Choquet integral we use a non-decreasing permutation $\mathbf{w}_\nearrow$. Then OWA is expressed as

$$C_v(\mathbf{x}) = \sum_{i=1}^{n} x_{\nearrow(i)} \left(Q(\frac{n-i+1}{n}) - Q(\frac{n-i}{n}) \right) = \sum_{i=1}^{n} x_{\searrow(i)} \left(Q(\frac{i}{n}) - Q(\frac{i-1}{n}) \right).$$

We also remind that $Q : [0,1] \to][0,1], Q(0) = 0, Q(1) = 1$ is a RIM quantifier which determines values of v as $v(\mathcal{A}) = Q\left(\frac{|\mathcal{A}|}{n}\right)$.

Proposition 2.143. *The Choquet integral with respect to a symmetric additive fuzzy measure is the arithmetic mean M, and the values of v are given by*

$$v(\mathcal{A}) = \frac{|\mathcal{A}|}{n}.$$

Proposition 2.144. *The Choquet integral with respect to a boolean fuzzy measure v coincides with the Sugeno integral (see Section 2.7 below).*

2- and 3-additive symmetric fuzzy measures

These are two special cases of symmetric fuzzy measures, for which we can write down explicit formulas for determination of the values of the fuzzy measure. By Proposition 2.142, Choquet integral with respect to any symmetric fuzzy measure is an OWA function with the weights $w_i = Q(\frac{i}{n}) - Q(\frac{i-1}{n})$. If v is additive, then $Q(t) = t$, and the Choquet integral becomes the arithmetic mean M.

Let us now determine function Q for less restrictive 2- and 3-additive fuzzy measures. These fuzzy measures allow interactions of pairs and triples of criteria, but not in bigger coalitions.

It turns out that in these two cases, the function Q is necessarily quadratic or cubic, as given in Proposition 2.72, p. 86, namely, for 2-additive symmetric fuzzy measure

$$Q(t) = at^2 + (1-a)t \text{ for some } a \in [-1, 1],$$

which implies that OWA weights are equidistant (i.e., $w_{i+1} - w_i = const$ for all $i = 1, \dots, n-1$). In the 3-additive case we have

$$Q(t) = at^3 + bt^2 + (1-a-b)t \text{ for some } a \in [-2, 4],$$

such that

- if $a \in [-2, 1]$ then $b \in [-2a-1, 1-a]$;
- if $a \in]1, 4]$ then $b \in [-3a/2 - \sqrt{3a(4-a)/4}, -3a/2 + \sqrt{3a(4-a)/4}]$.

2.6.6 Fitting fuzzy measures

Identification of the $2^n - 2$ values from the data (two are given explicitly as $v(\emptyset) = 0, v(N) = 1$) involves the least squares or least absolute deviation problems

$$\text{minimize} \sum_{k=1}^{K} \left(C_v(x_{1k}, \dots, x_{nk}) - y_k\right)^2, \text{ or}$$

$$\text{minimize} \sum_{k=1}^{K} \left|C_v(x_{1k}, \dots, x_{nk}) - y_k\right|,$$

subject to the conditions of monotonicity of the fuzzy measure (they translate into a number of linear constraints). If we use the indexing system outlined on p. 96, the conditions of monotonicity $v(\mathcal{T}) \leq v(\mathcal{S})$ whenever $\mathcal{T} \subseteq \mathcal{S}$ can be written as $v_i \leq v_j$ if $i \leq j$ and $AND(i,j) = i$ (AND is the usual bitwise operation, applied to the binary representations of i, j, which sets k-th bit of the result to 1 if and only if the k-th bits of i and j are 1).

Thus, in the least squares case we have the optimization problem

$$\begin{aligned} \text{minimize} \quad & \sum_{k=1}^{K} \left(< \mathbf{g}(x_{1k}, \ldots, x_{nk}), \mathbf{v} > - y_k\right)^2, \\ \text{s.t.} \quad & v_j - v_i \geq 0, \text{ for all } i < j \text{ such that } AND(i,j) = i, \\ & v_j \geq 0, j = 1, \ldots, 2^n - 2, v_{2^n-1} = 1, \end{aligned} \tag{2.74}$$

which is clearly a QP problem. In the least absolute deviation case we obtain

$$\begin{aligned} \text{minimize} \quad & \sum_{k=1}^{K} \left|< \mathbf{g}(x_{1k}, \ldots, x_{nk}), \mathbf{v} > - y_k\right|, \\ \text{s.t.} \quad & v_j - v_i \geq 0, \text{ for all } i < j \text{ such that } AND(i,j) = i, \\ & v_j \geq 0, j = 1, \ldots, 2^n - 2, v_{2^n-1} = 1, \end{aligned} \tag{2.75}$$

which is converted into an LP problem (see in Appendix A.2 how LAD is converted into LP). These are two standard problem formulations that are solved by standard QP and LP methods.

Note that in formulations (2.74) and (2.75) most monotonicity constraints will be redundant. It is sufficient to include only constraints such that $AND(i,j) = i$, i and j differ by only one bit (i.e., the cardinalities of the corresponding subsets satisfy $|\mathcal{J}| - |\mathcal{I}| = 1$). There will be $n(2^{n-1} - 1)$ non-redundant constraints. Explicit expression of the constraints in matrix form is complicated, but they are easily specified by using an incremental algorithm for each n. Further, many non-negativity constraints will be redundant as well (only $v(\{i\}) \geq 0, i = 1, \ldots, n$ are needed), but since they form part of a standard LP problem formulation anyway, we will keep them.

However, the main difficulty in these problems is the large number of unknowns, and typically a much smaller number of data [110, 189]. While modern LP and QP methods handle well the resulting (degenerate) problems, for $n \geq 10$ one needs to take into account the sparse structure of the system of constraints. For larger n (e.g., $n = 15$, $2^n - 1 = 32767$) QP methods are not as robust as LP, which can handle millions of variables. It is also important to understand that if the number of data $K \ll 2^n$, there will be multiple optimal solutions, i.e., (infinitely) many fuzzy measures that fit the data.

As discussed in [189], multiple solutions sometimes lead to counterintuitive results, because many values of v will be near 0 or 1. It was proposed to use a heuristic, which, in the absence of any data, chooses the "most additive" fuzzy measure, i.e., converts Choquet integral into the arithmetic mean. Grabisch [110, 116] has developed a heuristic least mean squares algorithm.

It is not difficult to incorporate the above mentioned heuristic into QP or LP problems like (2.74) and (2.75). Firstly, one may use the variables $\tilde{v}_j = v_j - \bar{v}_j$, instead of v_j ($\bar{v}_j$ denote the values of the additive symmetric fuzzy measure $v(\mathcal{A}) = \frac{|\mathcal{A}|}{n}$). In this case the default 0 values of the fitted $\tilde{v}_j$ (in the absence of data) will result in $v_j = \bar{v}_j$. On the other hand, it is possible to introduce penalty terms into the objective functions in (2.74) and (2.75) by means of artificial data, e.g., to replace the objective function in (2.74) with

$$\sum_{k=1}^{K} (< \mathbf{g}(x_{1k}, \dots, x_{nk}), \mathbf{v} > -y_k)^2 + \frac{p}{2^n} \sum_{k=1}^{2^n} (< \mathbf{g}(a_{1k}, \dots, a_{nk}), \mathbf{v} > -b_k)^2,$$

with p being a small penalty parameter, $\mathbf{a}_k$ being the vertices of the unit cube and $b_k = \frac{\sum_{i=1}^n a_{ik}}{n}$.

There are also alternative heuristic methods for fitting discrete Choquet integrals to empirical data. An overview of exact and approximate methods is provided in [110].

Other requirements on fuzzy measures

There are many other requirements that can be imposed on the fuzzy measure from the problem specifications. Some conditions are aimed at reducing the complexity of the fitting problem (by reducing the number of parameters), whereas others have direct meaningful interpretation.

Importance and interaction indices

The interaction indices defined in Section 2.6.4 are all linear functions of the values of the fuzzy measure. Conditions involving these functions can be expressed as linear equations and inequalities.

One can specify given values of importance (Shapley value) and interaction indices $\phi(i)$, I_{ij} (see pp. 105-106) by adding linear equality constraints to the problems (2.74) and (2.75). Of course, these values may not be specified exactly, but as intervals, say, for Shapley value we may have $a_i \le \phi(i) \le b_i$. In this case we obtain a pair of linear inequalities.

Substitutive criteria

For substitutive criteria i, j we add (see p. 108)

$$v(\mathcal{A} \cup \{i,j\}) \le v(\mathcal{A} \cup \{i\}) + (1 - \nu_{ij}) v(\mathcal{A} \cup \{j\}),$$
$$v(\mathcal{A} \cup \{i,j\}) \le v(\mathcal{A} \cup \{j\}) + (1 - \nu_{ij}) v(\mathcal{A} \cup \{i\}).$$

for all subsets $\mathcal{A} \subseteq \mathcal{N} \setminus \{i,j\}$, where $\nu_{ij} \in [0,1]$ is the degree of substitutivity. These are also linear inequality constraints added to the quadratic or linear programming problems.

k-additivity

Recall that Definition 2.121 specifies k-additive fuzzy measures through their Möbius transform

$$\mathcal{M}(\mathcal{A}) = 0$$

for any subset $\mathcal{A}$ with more than k elements. Since Möbius transform is a linear combination of values of v, we obtain a set of linear equalities. By using interaction indices, we can express k-additivity as (see Proposition 2.134) $I(\mathcal{A}) = 0$ for every $\mathcal{A} \subseteq \mathcal{N}$, $|\mathcal{A}| > k$, which is again a set of linear equalities.

All of the mentioned conditions on the fuzzy measures do not reduce the complexity of quadratic or linear programming problems (2.74) and (2.75). They only add a number of equality and inequality constraints to these problems. The aim of introducing these conditions is not to simplify the problem, but to better fit a fuzzy measure to the problem and data at hand, especially when the number of data is small.

Let us now consider simplifying assumptions, which do reduce the complexity. First recall that adding the symmetry makes Choquet integral an OWA function. In this case we only need to determine n (instead of $2^n - 2$) values. Thus we can use the techniques for fitting OWA weights, discussed in Section 2.5.5. In the case of 2- and 3-additive symmetric fuzzy measures we can fit generating functions Q, as discussed in Section 2.5.5.

λ-fuzzy measures

Fitting λ-fuzzy measures also involves a reduced set of parameters. Recall (p. 102) that λ-fuzzy measures are specified by n parameters $v(\{i\}), i = 1, \ldots, n$. The other values are determined from

$$v(\mathcal{A}) = \frac{1}{\lambda}\left(\prod_{i \in \mathcal{A}} (1 + \lambda v(\{i\})) - 1\right)$$

with the help of the parameter λ, which itself is computed from

$$\lambda + 1 = \prod_{i=1}^{n} (1 + \lambda v(\{i\})).$$

The latter is a non-linear equation, which is solved numerically on $(-1, 0)$ or $(0, \infty)$. This means that the Choquet integral C_v becomes a *nonlinear* function of parameters $v(\{i\})$, and therefore problems (2.74) and (2.75) become difficult non-linear programming problems. The problem of fitting these fuzzy measures was studied in [56, 135, 143, 153, 250, 253] and the methods are based on genetic algorithms.

Representation based on Möbius transformation

In this section we reformulate fitting problem for general k-additive fuzzy measures based on Möbius representation. Our goal is to use this representation to reduce the complexity of problems (2.74) and (2.75). We remind that this is an invertible linear transformation such that:

- $$\mathcal{M}(\mathcal{A}) = \sum_{\mathcal{B} \subseteq \mathcal{A}} (-1)^{|\mathcal{A} \setminus \mathcal{B}|} v(\mathcal{B}) \text{ and}$$
$$v(\mathcal{A}) = \sum_{\mathcal{B} \subseteq \mathcal{A}} \mathcal{M}(\mathcal{B}), \quad \text{for all } \mathcal{A} \subseteq \mathcal{N}.$$
- Choquet integral is expressed as
$$C_v(\mathbf{x}) = \sum_{\mathcal{A} \subseteq \mathcal{N}} \mathcal{M}(\mathcal{A}) \min_{i \in \mathcal{A}} x_i.$$
- k-additivity holds when
$$\mathcal{M}(\mathcal{A}) = 0$$
for any subset $\mathcal{A}$ with more than k elements.

As the variables we will use $m_j = m_{\mathcal{A}} = \mathcal{M}(\mathcal{A})$ such that $|\mathcal{A}| \leq k$ in some appropriate indexing system, such as the one based on cardinality ordering on p. 97. This is a much reduced set of variables ($\sum_{i=1}^{k} \binom{n}{i} - 1$ compared to $2^n - 2$). Now, monotonicity of a fuzzy measure, expressed as

$$v(\mathcal{A} \cup \{i\}) - v(\mathcal{A}) \geq 0, \quad \forall \mathcal{A} | i \notin \mathcal{A}, i = 1, \ldots, n,$$

converts into (2.54), and using k-additivity, into

$$\sum_{\mathcal{B} \subseteq A | i \in \mathcal{B}, |\mathcal{B}| \leq k} m_{\mathcal{B}} \geq 0, \quad \text{for all } \mathcal{A} \subseteq \mathcal{N} \text{ and all } i \in \mathcal{A}.$$

The (non-redundant) set of non-negativity constraints $v(\{i\}) \geq 0, i = 1, \ldots, n$, is a special case of the previous formula when $\mathcal{A}$ is a singleton, which simply become (see Note 2.81)

$$\sum_{\mathcal{B} = \{i\}} m_{\mathcal{B}} = m_{\{i\}} \geq 0, \; i = 1, \ldots, n.$$

Finally, condition $v(\mathcal{N}) = 1$ becomes $\sum\limits_{\mathcal{B} \subseteq \mathcal{N} || \mathcal{B}| \leq k} m_{\mathcal{B}} = 1.$

Then the problem (2.74) is translated into a simplified QP problem

$$\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{K}\left(\sum_{\mathcal{A}|\ |\mathcal{A}|\le k} h_{\mathcal{A}}(\mathbf{x}_j)m_{\mathcal{A}} - y_j\right)^2, \qquad (2.76)\\
\text{s.t.} \quad & \sum_{\mathcal{B}\subseteq A|i\in\mathcal{B},|\mathcal{B}|\le k} m_{\mathcal{B}} \ge 0,\\
& \qquad \text{for all } \mathcal{A}\subseteq\mathcal{N}, |\mathcal{A}|>1, \text{ and all } i\in\mathcal{A},\\
& m_{\{i\}} \ge 0,\ i=1,\dots,n,\\
& \sum_{\mathcal{B}\subseteq\mathcal{N}||\mathcal{B}|\le k} m_{\mathcal{B}} = 1,
\end{aligned}$$

where $h_{\mathcal{A}}(\mathbf{x}) = \min_{i\in\mathcal{A}} x_i$. Note that only the specified $m_{\mathcal{B}}$ are non-negative, others are unrestricted. The number of monotonicity constraints is the same for all k-additive fuzzy measures for $k = 2,\dots,n$. Similarly, a simplified LP problem is obtained from (2.75), with the same set of constraints as in (2.76).

A software package Kappalab provides a number of tools to calculate various quantities using fuzzy measures, such as the Möbius transform, interaction indices, k–additive fuzzy measures in various representations, and also allows one to fit values of fuzzy measures to empirical data. This package is available from `http://www.polytech.univ-nantes.fr/kappalab` and it works under R environment [145]. A set of C++ algorithms for the same purpose is available from `http://www.deakin.edu.au/~gleb/aotool.html`.

2.6.7 Generalized Choquet integral

Mesiar [179] has proposed a generalization of the Choquet integral, called Choquet-like integral.

Definition 2.145 (Generalized discrete Choquet integral). *Let $g : [0,1] \to [-\infty,\infty]$ be a continuous strictly monotone function. A generalized Choquet integral with respect to a fuzzy measure v is the function*

$$C_{v,g}(\mathbf{x}) = g^{-1}\left(C_v(g(\mathbf{x}))\right),$$

where C_v is the discrete Choquet integral with respect to v and $g(\mathbf{x}) = (g(x_1),\dots,g(x_n))$.

A special case of this construction was presented in [271].

$$C_{v,q}(\mathbf{x}) = \left(\sum_{i=1}^{n} x_{(i)}^q[v(H_i) - v(H_{i+1})]\right)^{1/q}, \quad q\in\Re. \qquad (2.77)$$

It is not difficult to see that this is equivalent to

$$C_{v,q}(\mathbf{x}) = \left(\sum_{i=1}^{n}\left[x_{(i)}^q - x_{(i-1)}^q\right]v(H_i)\right)^{1/q}. \qquad (2.78)$$

The generalized Choquet integral depends on the properties of the fuzzy measure v, discussed in this Chapter. For additive fuzzy measures it becomes a weighted quasi-arithmetic mean with the generating function g, and for symmetric fuzzy measures, it becomes a generalized OWA function, with the generating function g. Continuity of $C_{v,g}$ holds if $Ran(g) \neq [-\infty, \infty]$.

Fitting generalized Choquet integral to empirical data involves a modification of problems (2.74), (2.75) or (2.76), which consists in applying g to the components of $\mathbf{x}_k$ and y_k, (i.e., using the data $\{g(\mathbf{x}_k), g(y_k)\}$) provided g is known, or is fixed. In the case of fitting both g and v to the data, we use bi-level optimization, similar to that in Section 2.3.7.

The Choquet integral, as well as the Sugeno integral treated in the next section, are special cases of more general integrals with respect to a fuzzy measure. The interested reader is referred to [179, 210, 255].

2.7 Sugeno Integral

2.7.1 Definition and properties

Similarly to the Choquet integral, Sugeno integral is also frequently used to aggregate inputs, such as preferences in multicriteria decision making [80, 164]. Various important classes of aggregation functions, such as medians, weighted minimum and weighted maximum are special cases of Sugeno integral.

Definition 2.146 (Discrete Sugeno integral). *The Sugeno integral with respect to a fuzzy measure v is given by*

$$S_v(\mathbf{x}) = \max_{i=1,\ldots,n} \min\{x_{(i)}, v(H_i)\}, \tag{2.79}$$

where $\mathbf{x}_{\nearrow} = (x_{(1)}, x_{(2)}, \ldots, x_{(n)})$ is a non-decreasing permutation of the input $\mathbf{x}$, and $H_i = \{(i), \ldots, (n)\}$.

Sugeno integrals can be expressed, for arbitrary fuzzy measures, by means of the Median function (see Section 2.8.1 below) in the following way:

$$S_v(\mathbf{x}) = Med(x_1, \ldots, x_n, v(H_2), v(H_3), \ldots, v(H_n)).$$

Let us denote max by $\vee$ and min by $\wedge$ for compactness. We denote by $\mathbf{x} \vee \mathbf{y} = \mathbf{z}$ the componentwise maximum of $\mathbf{x}, \mathbf{y}$ (i.e., $z_i = \max(x_i, y_i)$), and by $\mathbf{x} \wedge \mathbf{y}$ their componentwise minimum.

Main properties

- Sugeno integral is a continuous idempotent aggregation function;

- An aggregation function is a Sugeno integral if and only if it is *min-homogeneous*, i.e., $S_v(x_1 \wedge r, \ldots, x_n \wedge r) = S_v(x_1, \ldots, x_n) \wedge r$ and *max-homogeneous*, i.e., $S_v(x_1 \vee r, \ldots, x_n \vee r) = S_v(x_1, \ldots, x_n) \vee r$ for all $\mathbf{x} \in [0,1]^n, r \in [0,1]$ (See [163], Th. 4.3. There are also alternative characterizations);
- Sugeno integral is *comonotone maxitive* and *comonotone minimitive*, i.e., $S_v(\mathbf{x} \vee \mathbf{y}) = S_v(\mathbf{x}) \vee S_v(\mathbf{y})$ and $S_v(\mathbf{x} \wedge \mathbf{y}) = S_v(\mathbf{x}) \wedge S_v(\mathbf{y})$ for all comonotone[27] $\mathbf{x}, \mathbf{y} \in [0,1]^n$;
- Sugeno integral is Lipschitz-continuous, with the Lipschitz constant 1 in any p–norm, which means it is a kernel aggregation function, see Definition 1.62, p. 23;
- The class of Sugeno integrals is closed under duality.

Calculation

Calculation of the discrete Sugeno integral is performed using Equation (2.79) similarly to calculating the Choquet integral on p. 96. We take the vector of pairs $((x_1, 1), (x_2, 2), \ldots, (x_n, n))$, where the second component of each pair is just the index i of x_i. The second component will help keeping track of all permutations.

Calculation of $S_v(\mathbf{x})$.

1. Sort the components of $((x_1, 1), (x_2, 2), \ldots, (x_n, n))$ with respect to the first component of each pair in non-decreasing order. We obtain $((x_{(1)}, i_1), (x_{(2)}, i_2), \ldots, (x_{(n)}, i_n))$, so that $x_{(j)} = x_{i_j}$ and $x_{(j)} \leq x_{(j+1)}$ for all i.
2. Let $\mathcal{T} = \{1, \ldots, n\}$, and $S = 0$.
3. For $j = 1, \ldots, n$ do
   a) $S := \max(S, \min(x_{(j)}, v(\mathcal{T})))$;
   b) $\mathcal{T} := \mathcal{T} \setminus \{i_j\}$
4. Return S.

Example 2.147. Let $n = 3$, let the values of v be given by

$$v(\emptyset) = 0,\ v(\mathcal{N}) = 1,\ v(\{1\}) = 0.5, v(\{2\}) = v(\{3\}) = 0,$$

$$v(\{1,2\}) = v(\{2,3\}) = 0.5,\ v(\{1,3\}) = 1.$$

and $\mathbf{x} = (0.8, 0.1, 0.6)$.

Step 1. We take $((0.8, 1), (0.1, 2), (0.6, 3))$.
Sort this vector of pairs to obtain $((0.1, 2), (0.6, 3), (0.8, 1))$.
Step 2. Take $\mathcal{T} = \{1, 2, 3\}$ and $S = 0$.
Step 3. a) $S := \max(0, \min(0.1, v(\{1, 2, 3\}))) = 0.1$;
b) $\mathcal{T} = \{1, 2, 3\} \setminus \{2\} = \{1, 3\}$;

[27] See footnote 18 on p. 95.

a) $S := \max(0.1, \min(0.6, v(\{1, 3\}))) = 0.6$;
b)$\mathcal{T} := \{1, 3\} \setminus \{3\} = \{1\}$;
a) $S := \max(0.6, \min(0.8, v(\{1\}))) = 0.6$.
Therefore $S_v(\mathbf{x}) = 0.6$.

Special cases

- The Sugeno integral with respect to a symmetric fuzzy measure given by $v(\mathcal{A}) = v(|\mathcal{A}|)$ is the Median $Med(x_1, \ldots, x_n, v(n-1), v(n-2), \ldots, v(1))$.

- Weighted maximum (max-min) $WMAX_{\mathbf{w}}(\mathbf{x}) = \max\limits_{i=1,\ldots,n} \min\{w_i, x_i\}$. An aggregation function is a weighted maximum if and only if it is the Sugeno integral with respect to a possibility measure;
- Weighted minimum (min-max) $WMIN_{\mathbf{w}}(\mathbf{x}) = \min\limits_{i=1,\ldots,n} \max\{w_i, x_i\}$. An aggregation function is a weighted minimum if and only if it is the Sugeno integral with respect to a necessity measure;
- Ordered weighted maximum $OWMAX_{\mathbf{w}}(\mathbf{x}) = \max\limits_{i=1,\ldots,n} \min\{w_i, x_{(i)}\}$ with a non-increasing weighting vector $1 = w_1 \geq w_2 \geq \ldots \geq w_n$. An aggregation function is an ordered weighted maximum if and only if it is the Sugeno integral with respect to a symmetric fuzzy measure. It can be expressed by means of the Median function as
$$OWMAX_{\mathbf{w}}(\mathbf{x}) = Med(x_1, \ldots, x_n, w_2, \ldots, w_n);$$
- Ordered weighted minimum $OWMIN_{\mathbf{w}}(\mathbf{x}) = \min\limits_{i=1,\ldots,n} \max\{w_i, x_{(i)}\}$ with a non-increasing weighting vector $w_1 \geq w_2 \geq \ldots \geq w_n = 0$. An aggregation function is an ordered weighted minimum if and only if it is the Sugeno integral with respect to a symmetric fuzzy measure. It can be expressed by means of the Median function as
$$OWMIN_{\mathbf{w}}(\mathbf{x}) = Med(x_1, \ldots, x_n, w_1, \ldots, w_{n-1});$$
- The Sugeno integral coincides with the Choquet integral if v is a boolean fuzzy measure.

Note 2.148. The weighting vectors in weighted maximum and minimum do not satisfy $\sum w_i = 1$, but $\max(\mathbf{w}) = 1$ and $\min(\mathbf{w}) = 0$ respectively.

Note 2.149. For the weighted maximum $WMAX_{\mathbf{w}}$ (v is a possibility measure) it holds $WMAX_{\mathbf{w}}(\mathbf{x} \vee \mathbf{y}) = WMAX_{\mathbf{w}}(\mathbf{x}) \vee WMAX_{\mathbf{w}}(\mathbf{y})$ for *all* vectors $\mathbf{x}, \mathbf{y} \in [0, 1]^n$ (a stronger property than comonotone maxitivity). For the weighted minimum $WMIN_{\mathbf{w}}$ (v is a necessity measure) it holds $WMIN_{\mathbf{w}}(\mathbf{x} \wedge \mathbf{y}) = WMIN_{\mathbf{w}}(\mathbf{x}) \wedge WMIN_{\mathbf{w}}(\mathbf{y})$ for all $\mathbf{x}, \mathbf{y} \in [0, 1]^n$.

Note 2.150. Ordered weighted maximum and minimum functions are related through their weights as follows: $OWMAX_{\mathbf{w}}(\mathbf{x}) = OWMIN_{\mathbf{u}}(\mathbf{x})$ if and only if $u_i = w_{i+1}, i = 1, \ldots, n-1$ [163].

Example 2.151. Let v be a fuzzy measure on $\{1,2\}$ defined by $v(\{1\}) = a \in [0,1]$ and $v(\{2\}) = b \in [0,1]$. Then

$$S_v(\mathbf{x}) = \begin{cases} x_1 \vee (b \wedge x_2), & \text{if } x_1 \le x_2, \\ (a \wedge x_1) \vee x_2, & \text{if } x_1 \ge x_2. \end{cases}$$

2.8 Medians and order statistics

2.8.1 Median

In statistics, the median of a sample is a number dividing the higher half of a sample, from the lower half. The median of a finite list of numbers can be found by arranging all the numbers in increasing or decreasing order and picking the middle one. If the number of inputs is even, one takes the mean of the two middle values.

The median is a type of average which is more representative of a “typical” value than the mean. It essentially discards very high and very low values (outliers). For example, the median price of houses is often reported in the real estate market, because the mean can be influenced by just one or a few very expensive houses, and will not represent the cost of a “typical” house in the area.

Definition 2.152 (Median). *The median is the function*

$$Med(\mathbf{x}) = \begin{cases} \frac{1}{2}(x_{(k)} + x_{(k+1)}), & \textit{if } n = 2k \textit{ is even} \\ x_{(k)}, & \textit{if } n = 2k-1 \textit{ is odd,} \end{cases}$$

where $x_{(k)}$ is the k-th largest (or smallest) component of $\mathbf{x}$.

Note 2.153. The median can be conveniently expressed as an OWA function with a special weighting vector. For an odd n let $w_{\frac{n+1}{2}} = 1$ and all other $w_i = 0$, and for an even n let $w_{\frac{n}{2}} = w_{\frac{n}{2}+1} = \frac{1}{2}$, and all other $w_i = 0$. Then $Med(\mathbf{x}) = OWA_{\mathbf{w}}(\mathbf{x})$.

Definition 2.154 (a-Median). *Given a value $a \in [0,1]$, the a-median is the function*

$$Med_a(\mathbf{x}) = Med(x_1, \ldots, x_n, \overbrace{a, \ldots, a}^{n-1 \; times}).$$

Note 2.155. a-medians are also the limiting cases of idempotent nullnorms, see Section 4.3. They have absorbing element a and are continuous, symmetric and associative (and, hence, bisymmetric). They can be expressed as

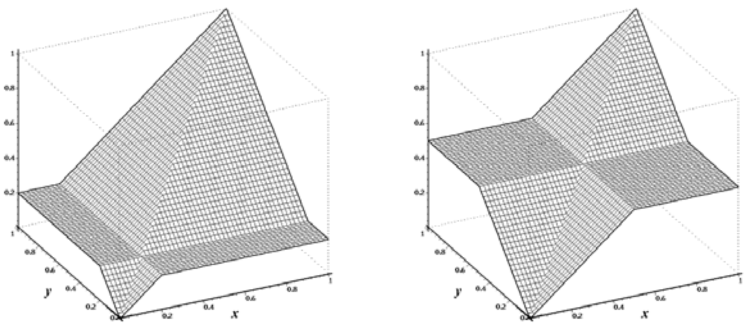


Fig. 2.25. 3D plots of a-medians $Med_{0.2}$ and $Med_{0.5}$.

$$Med_a(\mathbf{x}) = \begin{cases} \max(\mathbf{x}), & \text{if } \mathbf{x} \in [0,a]^n, \\ \min(\mathbf{x}), & \text{if } \mathbf{x} \in [a,1]^n, \\ a & \text{otherwise.} \end{cases}$$

An attractive property of the medians is that they are applicable to inputs given on the ordinal scale, i.e., when only the ordering, rather than the numerical values matter. For example, one can use medians for aggregation of inputs like labels of fuzzy sets, such as *very high*, *high*, *medium*, *low* and *very low*.

The concept of the weighted median was treated in detail in [268].

Definition 2.156 (Weighted median). *Let* $\mathbf{w}$ *be a weighting vector, and let* $\mathbf{u}$ *denote the vector obtained from* $\mathbf{w}$ *by arranging its components in the order induced by the components of the input vector* $\mathbf{x}$*, such that* $u_k = w_i$ *if* $x_i = x_{(k)}$ *is the k-th largest component of* $\mathbf{x}$*. The lower weighted median is the function*

$$Med_{\mathbf{w}}(\mathbf{x}) = x_{(k)}, \tag{2.80}$$

where k is the index obtained from the condition

$$\sum_{j=1}^{k-1} u_j < \frac{1}{2} \text{ and } \sum_{j=1}^{k} u_j \geq \frac{1}{2}. \tag{2.81}$$

The upper weighted median is the function (2.80) where k is the index obtained from the condition

$$\sum_{j=1}^{k-1} u_j \leq \frac{1}{2} \text{ and } \sum_{j=1}^{k} u_j > \frac{1}{2}.$$

Note 2.157. It is convenient to describe calculation of $Med_{\mathbf{w}}(\mathbf{x})$ using the following procedure. Take the vector of pairs $((x_1, w_1), (x_2, w_2), \dots, (x_n, w_n))$ and sort them in the order of decreasing x. We obtain $((x_{(1)}, u_1), (x_{(2)}, u_2), \dots, (x_{(n)}, u_n))$. Calculate the index k from the condition (2.81). Return $x_{(k)}$.

Note 2.158. The weighted median can be obtained by using penalty-based construction outlined on p. 298.

Main properties

The properties of the weighted median are consistent with averaging functions:

- Weighted median is a continuous idempotent aggregation function;
- If all the weights $w_i = \frac{1}{n}$, weighted median becomes the ordinary median Med;
- If any weight $w_i = 0$, then
$$Med_{\mathbf{w}}(\mathbf{x}) = Med_{(w_1,\dots,w_{i-1},w_{i+1},\dots,w_n)}(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n),$$
i.e., the input x_i can be dropped from the aggregation procedure;
- If any input value is repeated, one can use just a copy of this value and add the corresponding weights, namely if $x_i = x_j$ for some $i < j$, then
$$Med_{\mathbf{w}}(\mathbf{x}) = Med_{\tilde{\mathbf{w}}}(x_1, \dots, x_{j-1}, x_{j+1}, \dots, x_n),$$
where $\tilde{\mathbf{w}} = (w_1, \dots, w_{i-1}, w_i + w_j, w_{i+1}, \dots, w_{j-1}, w_{j+1}, \dots, w_n)$.

As far as learning the weights of weighted medians from empirical data, Yager [268] presented a gradient based local optimization algorithm. Given that such a method does not guarantee the globally optimal solution, it is advisable to combine it with a generic global optimization scheme, such as multistart local search or simulated annealing.

Based on the weighted median, Yager [268] also defined an ordinal OWA function, using the following construction. We recall (see Section 2.5) that $OWA_{\mathbf{w}}(\mathbf{x}) =< \mathbf{w}, \mathbf{x}_{\searrow} >$, i.e., the weighted mean of the vector $\mathbf{x}_{\searrow}$. By replacing the weighted mean with weighted median we obtain

---

Definition 2.159 (Ordinal OWA). *The ordinal OWA function is*
$$OOWA_{\mathbf{w}}(\mathbf{x}) = Med_{\mathbf{w}}(\mathbf{x}_{\searrow}).$$

Note 2.160. Since the components of the argument of the weighted median in Definition 2.159 are already ordered, calculation of the ordinal OWA is reduced to the formula
$$OOWA_{\mathbf{w}}(\mathbf{x}) = x_{(k)},$$
where k is the index obtained from the condition
$$\sum_{j=1}^{k-1} w_j < \frac{1}{2} \text{ and } \sum_{j=1}^{k} w_j \geq \frac{1}{2}.$$

A more general class of aggregation functions on an ordinal scale is that of weighted ordinal means, presented in [148].

2.8.2 Order statistics

Definition 2.161 (Order statistic). *The k-th order statistic is the function*

$$kOS(\mathbf{x}) = x_{(k)},$$

i.e., its value the k-th ***smallest***[28] *component of* $\mathbf{x}$.

Note 2.162. The order statistics can be conveniently expressed as OWA functions with special weighting vectors. Let $\mathbf{w} = (0, 0, \ldots, 0, 1, 0 \ldots, 0)$, i.e., $w_i = 0$ for $i \neq n - k + 1$ and $w_{n-k+1} = 1$. Then $kOS(\mathbf{x}) = OWA_{\mathbf{w}}(\mathbf{x})$.

2.9 Key references

P.S. Bullen. *Handbook of Means and Their Inequalities.* Kluwer Academic Publishers, Dordrecht, 2003.

D. Denneberg. *Non-additive Measure and Integral.* Kluwer, Dordrecht, 1994.

J. Fodor and M. Roubens. *Fuzzy Preference Modelling and Multicriteria Decision Support.* Kluwer, Dordrecht, 1994.

M. Grabisch, H.T. Nguyen, and E.A. Walker. *Fundamentals of Uncertainty Calculi, with Applications to Fuzzy Inference.* Kluwer, Dordrecht, 1995.

M. Grabisch, T. Murofushi, and M. Sugeno, editors, *Fuzzy Measures and Integrals. Theory and Applications.*, Physica-Verlag, Heidelberg, 2000.

Y. Torra, V. Narukawa. *Modeling Decisions. Information Fusion and Aggregation Operators.* Springer, Berlin, Heidelberg, 2007.

Z. Wang and G. Klir. *Fuzzy Measure Theory.* Plenum Press, New York, 1992.

R. Yager and D. Filev. *Essentials of Fuzzy Modelling and Control.* J. Wiley & Sons, New York, 1994.

R. Yager and J. Kacprzyk, editors. *The Ordered Weighted Averaging Operators. Theory and Applications.* Kluwer, Boston, 1997.

---

[28] Note that in OWA, weighted median and ordinal OWA, $x_{(k)}$ denotes the k-th **largest** element of $\mathbf{x}$.

3

Conjunctive and Disjunctive Functions

3.1 Semantics

As their names imply, conjunctive aggregation functions model conjunction (i.e., the logical AND), and disjunctive aggregation functions model disjunction (logical OR). Consider the situation where various fuzzy criteria are aggregated as in

If t_1 *is* A_1 **AND** t_2 *is* A_2 **AND** $\ldots$ t_n *is* A_n **THEN ...**

Conjunction does not allow for compensation: low scores for some criteria cannot be compensated by other scores. If to obtain a driving license one has to pass both theory and driving tests, no matter how well one does in the theory test, it does not compensate for failing the driving test.

Thus it is the smallest value of any of the inputs which bounds the output value (from above). Hence we have the definition

Definition 3.1 (Conjunctive aggregation). *An aggregation function f has conjunctive behavior (or is conjunctive) if for every $\mathbf{x}$ it is bounded by*

$$f(\mathbf{x}) \leq \min(\mathbf{x}) = \min(x_1, x_2, \ldots, x_n).$$

For disjunctive aggregation functions we have exactly the opposite. Satisfaction of any of the criteria is enough by itself, although more than one positive input pushes the total up. For example, both a wide open door (when you come home) and the sound of the alarm are indicators of a burglary, and either one is sufficient to raise suspicion. We may formalize this in a logical statement "If the door is open OR the alarm sounds, THEN it may be a burglary". But if both happen at the same time, they reinforce each other, and our suspicion is stronger than the suspicion caused by any of these indicators by itself (i.e., greater than their maximum). We define disjunctive aggregation as

Definition 3.2 (Disjunctive aggregation). *An aggregation function f has disjunctive behavior (or is disjunctive) if for every $\mathbf{x}$ it is bounded by*

$$f(\mathbf{x}) \geq \max(\mathbf{x}) = \max(x_1, x_2, \ldots, x_n).$$

3.2 Duality

Recall from Chapter 1 that given a strong negation N (Definition 1.48), any aggregation function f has an N-dual aggregation function f_d associated to it (Definition 1.54), and that conjunctive and disjunctive aggregation functions are dual to each other:

Proposition 3.3. *If f is a conjunctive aggregation function, its N-dual f_d (with respect to any strong negation N) is a disjunctive aggregation function, and vice versa.*

Typically the standard negation $N(t) = 1 - t$ is used, and then f_d is called simply dual of f, although one is not restricted to this choice.

Example 3.4. The dual of the minimum is maximum. The dual of the product T_P is the dual product S_P (also called probabilistic sum)

$$T_P(\mathbf{x}) = \prod_{i=1}^{n} x_i, \quad S_P(\mathbf{x}) = 1 - \prod_{i=1}^{n}(1 - x_i).$$

Duality is very convenient, as it allows one to study only conjunctive aggregation functions, and then obtain the analogous results for disjunctive functions by duality.

Note 3.5. Some aggregation functions are self-dual, i.e., $f(\mathbf{x}) = N(f(N(\mathbf{x})))$. We study them in detail in Chapter 4. However, there are no self-dual conjunctive or disjunctive aggregation functions.

3.3 Generalized OR and AND – functions

It follows from their definitions that 0 is the absorbing element of a conjunctive aggregation function, and 1 is the absorbing element of a disjunctive aggregation function.

Conjunctive and disjunctive aggregation may or may not be symmetric. A neutral element may or may not exist. If it exists, then $e = 1$ for conjunctive aggregation and $e = 0$ for disjunctive. But if the neutral element is $e = 1$, then the aggregation function is necessarily conjunctive. Similarly, if the neutral element is $e = 0$, then the aggregation function is necessarily disjunctive. The existence of a neutral element seems quite logical, and is postulated in many studies.

Definition 3.6 (Semicopula). *An aggregation function f is called a semicopula, if it has the neutral element $e = 1$. Its dual (an aggregation function with the neutral element $e = 0$) is called dual semicopula.*

Evidently, semicopulas are conjunctive, and their duals are disjunctive.

The prototypical conjunctive and disjunctive aggregation functions are the *minimum* and *maximum*, they are the limiting cases, and form the boundary with the averaging functions.

Example 3.7. Aggregation function $f(x_1, x_2) = x_1 x_2$ is conjunctive, symmetric, and has neutral element $e = 1$. It is a semi-copula. Aggregation function $f(x_1, x_2) = x_1^2 x_2^2$ is also conjunctive and symmetric, but it has no neutral element. The function $f(x_1, x_2) = x_1 x_2^2$ is conjunctive, asymmetric, and has no neutral element.

Consider now the Lipschitz property (see Definition 1.58), and let us concentrate on 1-Lipschitz aggregation functions, Definition 1.61.

Definition 3.8 (Quasi-copula). *An aggregation function f is called a quasi-copula, if it is 1-Lipschitz, and has neutral element $e = 1$.*

Evidently, each quasi-copula is a semicopula, and hence conjunctive, but not the other way around.

- Conjunctive aggregation functions, semicopulas and quasi-copulas form convex classes. That is, any convex combination of two aggregation functions f_1, f_2 from one class also belongs to that class

$$f = [\alpha f_1 + (1 - \alpha) f_2] \in \mathcal{F}, \text{ if } f_1, f_2 \in \mathcal{F} \text{ and } \alpha \in [0, 1].$$

- A pointwise minimum or maximum of two conjunctive aggregation functions, semicopulas or quasi-copulas also belongs to the same class. [1]

Another useful property is monotonicity with respect to argument cardinality.

Definition 3.9 (Monotonicity with respect to argument cardinality). *An extended aggregation function F is monotone non-increasing with respect to argument cardinality, if*

$$f_n(x_1, \ldots, x_n) \geq f_{n+1}(x_1, \ldots, x_n, x_{n+1}),$$

for all $n > 1$ and any $x_1, \ldots, x_{n+1} \in [0, 1]$. F is monotone non-decreasing with respect to argument cardinality if the sign of the inequality is reversed. If F is symmetric, then the positions of the inputs do not matter.

[1] Pointwise minimum of two functions f,g means $\min(f, g)(\mathbf{x}) = \min(f(\mathbf{x}), g(\mathbf{x}))$ for all $\mathbf{x}$ in their (common) domain. Pointwise maximum is defined similarly.

Of course, in general members of the family F need not be related, and thus extended conjunctive functions do not necessarily verify the condition of Definition 3.9, but it is reasonable to expect that by adding new inputs to an extended conjunctive aggregation function, the output can only become smaller. In the same way, it is reasonable to expect that extended disjunctive aggregation functions are monotone non-decreasing with respect to argument cardinality, i.e., adding inputs can only reinforce the output [2].

3.4 Triangular norms and conorms

Two special and well–known classes of conjunctive and disjunctive aggregation functions are the *triangular norms and conorms.*[3] Triangular norms were originally introduced by Menger [178] as operations for the fusion of distribution functions needed by triangle inequality generalization of a metric on statistical metric spaces. Menger's triangular norms formed a large, rather heterogeneous class of symmetric bivariate aggregation functions fulfilling $f(1, a) > 0$ whenever $a > 0$. Nowadays the definition of triangular norms due to Schweizer and Sklar [221] includes associativity and the neutral element $e = 1$. Note that associativity allowed the extension of triangle inequality to the polygonal inequality, including the fact, that now triangular norms can be applied to any finite number of inputs, that is, they form extended aggregation functions as in Definition 1.6. Triangular norms have become especially popular as models for fuzzy sets intersection. They are also applied in studies of probabilistic metric spaces, many-valued logic, non-additive measures and integrals, etc. For an exhaustive state-of-the-art overview in the field of triangular norms we recommend the recent monographs [6, 142].

3.4.1 Definitions

Definition 3.10 (Triangular norm (t–norm)). *A triangular norm (t–norm for short) is a bivariate aggregation function* $T : [0, 1]^2 \to [0, 1]$*, which is associative, symmetric and has neutral element* 1*.*

It follows immediately that a t-norm is a conjunctive aggregation function. Its dual (hence disjunctive) aggregation function is called a triangular conorm.

Definition 3.11 (Triangular conorm (t–conorm)). *A triangular conorm (t–conorm for short) is a bivariate aggregation function* $S : [0, 1]^2 \to [0, 1]$*, which is associative, symmetric and has neutral element* 0*.*

[2] Consider adding pieces of positive evidence in a court trial. Of course, there could be negative evidence, in which case the aggregation is of mixed type, see Chapter 4.

[3] Triangular conorms are also known as *s-norms* , see e.g., [6].

As with any conjunctive and disjunctive aggregation function, each t–norm T and each t–conorm S have respectively 0 and 1 as absorbing elements.

Example 3.12. The four basic examples of t–norms and t–conorms are the following:

1. $T_{min}(x, y) = \min(x, y)$ (minimum) $S_{max}(x, y) = \max(x, y)$ (maximum)
2. $T_P(x, y) = x \cdot y$ (product) $S_P(x, y) = x + y - x \cdot y$ (probabilistic sum)
3. $T_L(x, y) = \max(0, x + y - 1)$ $S_L(x, y) = \min(1, x + y)$ (Łukasiewicz t–norm) (Łukasiewicz t–conorm)
4. $$T_D(x, y) = \begin{cases} 0, & \text{if } (x, y) \in [0, 1[^2 \\ \min(x, y) & \text{otherwise} \end{cases} \quad \text{(drastic product)} \tag{3.1}$$

   and

   $$S_D(x, y) = \begin{cases} 1, & \text{if } (x, y) \in]0, 1]^2 \\ \max(x, y) & \text{otherwise.} \end{cases} \quad \text{(drastic sum)} \tag{3.2}$$

Note 3.13. Other notations that can be found in the literature (e.g., [6, 221]) are the following:

- $P, Prod, \Pi$ and $P^*, Prod^*, \Pi^*$ for the product and the dual product.
- W and W^* for the Łukasiewicz t–norm and the Łukasiewicz t–conorm.
- Z and Z^* for the drastic product and the drastic sum.

The product and Łukasiewicz t–norms are prototypical examples of two important subclasses of t–norms, namely, strict and nilpotent t–norms, which are studied in detail in Section 3.4.3. The graphs of these four basic examples are presented on Figures 3.1 and 3.2.

In the discussion of the properties of t–norms and t–conorms, we will pay particular attention to t–norms, and will obtain similar results for t–conorms by duality.

A related class of functions is called *triangular subnorms.*

Definition 3.14 (Triangular subnorm (t-subnorm)). *A t–subnorm is a function $f : [0, 1]^2 \to [0, 1]$, which is non-decreasing, associative, symmetric and conjunctive.*

Evidently, any t–norm is a t-subnorm but not vice–versa. For instance the zero function is a t-subnorm but not a t–norm. t-subnorms are in general not aggregation functions because $f(1, 1) = 1$ may not hold. One can always construct a t–norm from any t–subnorm by just enforcing the neutral element, i.e., by re-defining its values on the set $\{(x, y) \in [0, 1]^2 : x = 1 \text{ or } y = 1\}$. Of course, the resulting t–norm would generally be discontinuous. The drastic product is an example of such a construction, where the t–subnorm was the zero function.

The dual of a t–subnorm is called a *t–superconorm.*

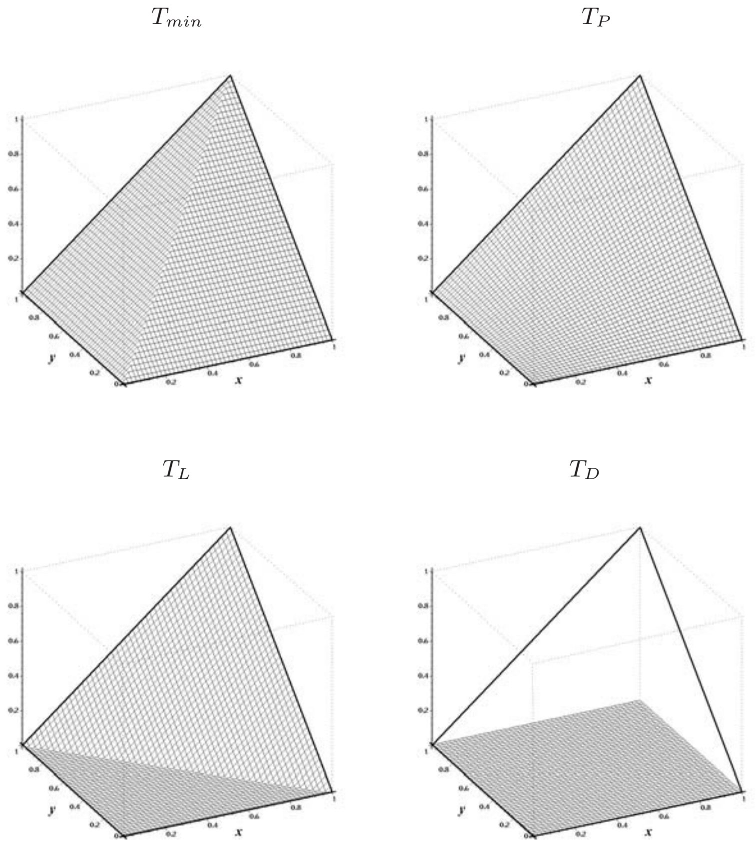


Fig. 3.1. The four basic t–norms: the minimum, product, Łukasiewicz and drastic product.

Definition 3.15 (Triangular superconorm (t-superconorm)). *A t–superconorm is a function* $f : [0,1]^2 \to [0,1]$*, which is non-decreasing, associative, symmetric and disjunctive.*

Evidently, any t–conorm is a t-superconorm but not vice–versa. We can obtain a t–conorm from any t–superconorm by enforcing the neutral element 0, for example, the drastic sum was obtained in this way from the function $f(x, y) = 1$.

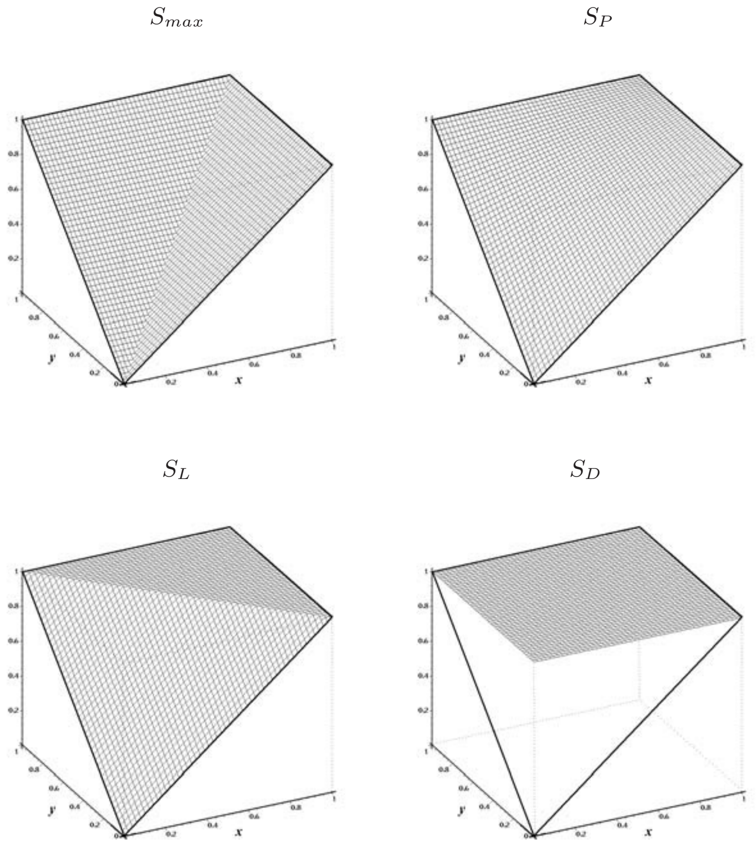


Fig. 3.2. The four basic t–conorms: the maximum, probabilistic sum, Łukasiewicz and drastic sum.

3.4.2 Main properties

Because of their associativity, t-norms are defined for any number of arguments $n \geq 1$ (with the usual convention $T(t) = t$), hence they are extended aggregation functions. Indeed, the associativity implies

$$T(x_1, x_2, x_3) = T(x_1, T(x_2, x_3)) = T(T(x_1, x_2), x_3),$$

and, in general,

$$T(x_1, x_2, \ldots, x_n) = T(x_1, T(x_2, \ldots, x_n)) = \ldots = T(T(x_1, \ldots, x_{n-1}), x_n).$$

Example 3.16. The n-ary forms of min and product are obvious. For Łukasiewicz and the drastic product we have the following extensions,

$$T_L(x_1, \ldots, x_n) = \max(\sum_{i=1}^{n} x_i - (n-1), 0),$$

$$T_D(x_1, \ldots, x_n) = \begin{cases} x_i, \text{ if } x_j = 1 \text{ for all } j \neq i, \\ 0 \quad \text{otherwise.} \end{cases}$$

We can derive the following properties by some basic algebra.

- Every t–norm satisfies the boundary conditions: for each $t \in [0, 1]$:
$$T(t, 0) = T(0, t) = 0, \tag{3.3}$$
$$T(t, 1) = T(1, t) = t, \tag{3.4}$$
which are just restatements of the fact that a t–norm has absorbing element 0 and neutral element 1.
- The weakest and the strongest t–norm are the *drastic product* and the *minimum*: $T_D(\mathbf{x}) \leq T(\mathbf{x}) \leq \min(\mathbf{x})$ for every $\mathbf{x} \in [0, 1]^n$ and every t–norm T.
- The only idempotent t–norm is the *minimum.*
- From their definitions we can deduce that
$$T_D(\mathbf{x}) \leq T_L(\mathbf{x}) \leq T_P(\mathbf{x}) \leq \min(\mathbf{x})$$
for every $\mathbf{x} \in [0, 1]^n$.
- t–norms are monotone non-increasing with respect to argument cardinality
$$T_{n+1}(x_1, \ldots, x_{n+1}) \leq T_n(x_1, \ldots, x_n) \text{ for all } n > 1.$$
- t–norms may or may not be continuous. The drastic product is an example of a discontinuous t–norm. Minimum is a continuous t–norm.
- The Lipschitz constant of any Lipschitz t–norm is at least one: $M \geq 1$ (see Definition 1.58).
- Not all t–norms are comparable (Definition 1.56).
- A pointwise minimum or maximum [4] of two t–norms is not generally a t–norm, although it is a conjunctive aggregation function.
- A linear combination of t–norms $aT_1(\mathbf{x}) + bT_2(\mathbf{x})$, $a, b \in \Re$, is not generally a t–norm, although it is a conjunctive aggregation function if $a, b \in [0, 1], b = 1 - a$.

[4] See footnote 1, p. 125.

The equivalent properties of t–conorms are listed below.

- Every t–conorm satisfies the boundary conditions: for each $t \in [0,1]$:

$$S(t,1) = S(1,t) = 1, \tag{3.5}$$

$$S(t,0) = S(0,t) = t, \tag{3.6}$$

  which are just restatements of the fact that a t–conorm has absorbing element 1 and neutral element 0.
- The weakest and the strongest t–conorm are the *maximum* and the *drastic sum*: $\max(\mathbf{x}) \leq S(\mathbf{x}) \leq S_D(\mathbf{x})$ for every $\mathbf{x} \in [0,1]^n$ and every t–conorm S.
- The only idempotent t–conorm is the *maximum*.
- From their definitions we can deduce that

$$\max(\mathbf{x}) \leq S_P(\mathbf{x}) \leq S_L(\mathbf{x}) \leq S_D(\mathbf{x})$$

  for every $\mathbf{x} \in [0,1]^n$.
- t–conorms are monotone non-decreasing with respect to argument cardinality

$$S_{n+1}(x_1, \ldots, x_{n+1}) \geq S_n(x_1, \ldots, x_n) \text{ for all } n > 1.$$

- t–conorms may or may not be continuous. The drastic sum is an example of a discontinuous t–conorm. Maximum is a continuous t–conorm.
- The Lipschitz constant of any Lipschitz t–conorm is at least one: $M \geq 1$.
- Not all t–conorms are comparable.
- A pointwise minimum or maximum of two t–conorms is not generally a t–conorm, although it is a disjunctive aggregation function.
- A linear combination of t–conorms $aS_1(\mathbf{x}) + bS_2(\mathbf{x})$, $a, b \in \Re$, is not generally a t–conorm, although it is a disjunctive aggregation function if $a, b \in [0,1], b = 1 - a$.

The n-ary forms of the maximum and probabilistic sum t–conorms are obvious. In the case of Łukasiewicz and drastic sum we have the following formulae:

$$S_L(x_1, \ldots, x_n) = \min(\sum_{i=1}^{n} x_i, 1)$$

$$S_D(x_1, \ldots, x_n) = \begin{cases} x_i, \text{ if } x_j = 0 \text{ for all } j \neq i, \\ 1 \quad \text{otherwise.} \end{cases}$$

We recall the definitions of zero and one divisors (Definitions 1.34 and 1.37), which for t–norms and t-conorms take the special form

Definition 3.17 (Zero and one divisors). *An element $a \in]0,1[$ is a zero divisor of a t–norm T if there exists some $b \in]0,1[$ such that $T(a,b) = 0$.*

An element $a \in]0,1[$ is a one divisor of a t–conorm S if there exists some $b \in]0,1[$ such that $S(a,b) = 1$.

Definition 3.18 (Nilpotent element). *An element $a \in]0,1[$ is a nilpotent element of a t–norm T if there exists an $n \in \{1,2,\ldots\}$ such that*

$$T_n(\overbrace{a,\ldots,a}^{n-times}) = 0.$$

An element $b \in]0,1[$ is a nilpotent element of a t–conorm S if there exists an $n \in \{1,2,\ldots\}$ such that

$$S_n(\overbrace{b,\ldots,b}^{n-times}) = 1.$$

Any $a \in]0,1[$ is a nilpotent element and also a zero divisor of the Łukasiewicz t–norm T_L as well as of the drastic product. The minimum and the product t-norms have neither nilpotent elements nor zero divisors.

Note 3.19. 1. Each nilpotent element a of a t–norm T is also a zero divisor of T, but not conversely. For that, it is enough to consider the *nilpotent minimum* defined by

$$T_{nM}(x,y) = \begin{cases} 0, & \text{if } x+y \le 1, \\ \min(x,y) & \text{otherwise }, \end{cases}$$

whose set of nilpotent elements is the interval $]0,0.5]$ and its set of zero divisors is the interval $]0,1[$.

2. If a is a nilpotent element (a zero divisor) of a t–norm T then each $b \in]0,a[$ is also a nilpotent element (zero divisor) of T.

Proposition 3.20. *Any t–norm has zero divisors if and only if it has nilpotent elements.*

3.4.3 Strict and nilpotent t–norms and t–conorms

Definition 3.21. ***(Strict t–norm)*** *A t–norm T is called strict if it is continuous and strictly monotone on $]0,1]^2$, i.e., $T(t,u) < T(t,v)$ whenever $t > 0$ and $u < v$.*

Note 3.22. Of course, there are no t–norms (or any conjunctive aggregation functions) strictly monotone on the whole domain (as they have an absorbing element $f(t,0) = f(0,t) = 0$ for all $t \in [0,1]$). Thus strict monotonicity is relaxed to hold only for $x_i > 0$.

Note 3.23. A strictly increasing t-norm (on $]0,1]^n$) need not be strict (it can be discontinuous).

Definition 3.24. ***(Nilpotent t–norm)*** *A t–norm T is called nilpotent if it is continuous and each element $a \in]0,1[$ is a nilpotent element of T, i.e., if there exists an $n \in \{1,2,\ldots\}$ such that $T(\overbrace{a,\ldots,a}^{n-times}) = 0$ for any $a \in]0,1[$.*

Example 3.25. The product T_P is a strict t–norm and the Łukasiewicz T_L is a nilpotent t–norm.

Of course, there are t–norms that are neither strict nor nilpotent. However, a t–norm cannot be at the same time strict and nilpotent.

Example 3.26. The drastic product is a non-continuous t–norm for which each element of $]0,1[$ is nilpotent. It is an example of t–norm that is neither strict nor nilpotent (because it is discontinuous). The minimum is a continuous t–norm which is neither strict nor nilpotent. The following t–norm is strictly monotone on $]0,1]^2$ but it is discontinuous, and hence it is not strict

$$T(x,y) = \begin{cases} \frac{xy}{2}, & \text{if } (x,y) \in [0,1[^2 \\ \min(x,y) & \text{otherwise.} \end{cases}$$

For triangular conorms we have similar definitions, essentially obtained by duality.

Definition 3.27. ***(Strict t–conorm)*** *A t–conorm S is called strict if it is continuous and strictly increasing on $[0,1[^2$, i.e., $S(t,u) < S(t,v)$ whenever $t < 1$ and $u < v$.*

Note 3.28. Of course, there are no t–conorms (or any disjunctive aggregation functions) strictly increasing on the whole domain (as they have an absorbing element $f(t,1) = f(1,t) = 1$ for all $t \in [0,1]$). Thus strict monotonicity is relaxed to hold only on $[0,1[^n$.

Definition 3.29. ***(Nilpotent t–conorm)*** *A t–conorm S is called nilpotent if it is continuous and each element $a \in]0,1[$ is a nilpotent element of S, i.e., if there exists an $n \in \{1,2,\ldots\}$ such that $S(\overbrace{a,\ldots,a}^{n-times}) = 1$, for any $a \in]0,1[$.*

Example 3.30. The probabilistic sum S_P is a strict t–conorm and the Łukasiewicz S_L is a nilpotent t–conorm. But the drastic sum and the maximum t–conorms are neither strict nor nilpotent t–conorms.

Note 3.31. A t–conorm S is strict (nilpotent) if and only if its dual t–norm T is strict (nilpotent).

3.4.4 Archimedean t–norms and t–conorms

Definition 3.32. *(**Archimedean t–norm**) A t–norm is called Archimedean if for each $(a,b) \in]0,1[^2$ there is an $n \in \{1,2,\ldots\}$ with $T(\overbrace{a,\ldots,a}^{n-times}) < b$.*

This property is equivalent to the limit property, i.e., for all $t \in]0,1[$: $\lim\limits_{n\to\infty} T_n(\overbrace{t,\ldots,t}^{n-times}) = 0$. It also implies that the only idempotent elements (i.e., the values a such that $T(a,a) = a$) 0 and 1.

Definition 3.33. *(**Archimedean t–conorm**) A t–conorm is called Archimedean if for each $(a,b) \in]0,1[^2$ there is an $n \in \{1,2,\ldots\}$ with $S(\overbrace{a,\ldots,a}^{n-times}) > b$.*

Proposition 3.34. *A continuous Archimedean t–norm (t–conorm) is either strict or nilpotent.*

- *If a t–norm T (t–conorm S) is strict then it is Archimedean;*
- *If a t–norm T (t–conorm S) is nilpotent then it is Archimedean.*

Archimedean t–norms are usually continuous, although there are some examples when they are discontinuous. We will concentrate on continuous t–norms as they play an important role in applications. One special property of a continuous Archimedean t–norm (or t–conorm) is that it is strictly increasing, except for the subset where its value is 0 (or 1 for t-conorms).

Proposition 3.35. *A continuous t–norm T is Archimedean if and only if it is strictly increasing on the subset $\{(x,y) \in [0,1]^2 | \; T(x,y) > 0\}$.*

Note 3.36. In [142], p.27 the authors call this property the conditional cancelation law: $\forall x,y,z, T(x,y) = T(x,z) > 0$ implies $y = z$.

As we shall see in the subsequent sections, continuous Archimedean t–norms are very useful for the following reasons: a) they form a dense subset in the set of all continuous t–norms, and b) they can be represented via additive and multiplicative generators. This representation reduces calculation of a t–norm (a multivariate function) to calculation of the values of its univariate generators.

3.4.5 Additive and multiplicative generators

In this section we express t–norms and t–conorms by means of a single real function $g : [0,1] \to [0,\infty]$ with some specific properties.

Let g be a strictly decreasing bijection. The inverse of g exists and is denoted by g^{-1}, and of course $(g^{-1} \circ g)(t) = (g \circ g^{-1})(t) = t$. Also *Dom* g = *Ran* g^{-1} and *Dom* g^{-1} = *Ran* g. This is very convenient for our purposes, since *Dom* $g^{-1} = [0,\infty]$, and we intend to apply g^{-1} to a construction involving the values $g(x_1), g(x_2), \ldots$, namely (see Fig. 3.4)

$$g^{-1}(g(x_1) + g(x_2) + \ldots + g(x_n)),$$

so the argument of g^{-1} can be any non-negative value.

However, we also need a well defined inverse for the case when $g : [0,1] \to [0,a]$ is a strictly decreasing bijection[5], and $0 < a < \infty$. Here again, the inverse g^{-1} exists, but *Dom* $g^{-1} = [0,a]$, see Fig. 3.3. We want to have the flexibility to use g^{-1} with any non-negative argument, so we extend the domain of g^{-1} by using

$$g^{(-1)}(t) = \begin{cases} g^{-1}(t), & \text{if } t \in [0,a], \\ 0 & \text{otherwise.} \end{cases}$$

We call the resulting function pseudo-inverse[6]. Note that we can express $g^{(-1)}$ as

$$g^{(-1)}(t) = \sup\{z \in [0,1] \mid g(z) > t\}$$

for all $t \in [0,\infty]$. This definition covers both cases $a < \infty$ and $a = \infty$, i.e., applies to any continuous strictly decreasing function $g : [0,1] \to [0,\infty]$ verifying $g(1) = 0$.

The reason why we concentrated on continuous strictly decreasing functions is that any continuous Archimedean t-norm can be represented with the help of a continuous *additive generator*. An additive generator is a strictly decreasing function $g : [0,1] \to [0,\infty]$ verifying $g(1) = 0$.

Proposition 3.37. *Let T be a continuous Archimedean t–norm. Then it can be written as*

$$T(x,y) = g^{(-1)}(g(x) + g(y)), \tag{3.7}$$

where $g : [0,1] \to [0,\infty]$, $g(1) = 0$, is a continuous strictly decreasing function, called an ***additive generator*** *of T.*

Note 3.38. For more than two arguments we have

$$T(\mathbf{x}) = g^{(-1)}(g(x_1) + g(x_2) + \ldots + g(x_n)).$$

[5] Note that since g is a bijection, it verifies $g(0) = a, g(1) = 0$. We also remind that a strictly monotone bijection is always continuous.

[6] Section 3.4.6 treats pseudo-inverses in the general case, when g is not necessarily continuous, strictly decreasing or a bijection.

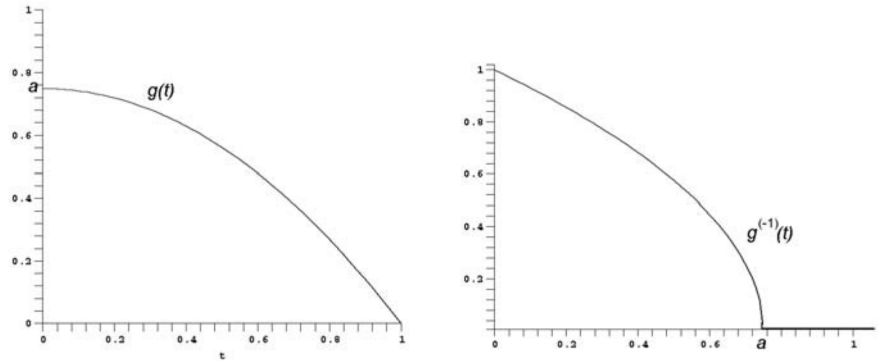


Fig. 3.3. A typical additive generator of a nilpotent t–norm and its pseudo-inverse.

Note 3.39. The converse of Proposition 3.37 is also true: Given any strictly decreasing function $g : [0,1] \to [0,\infty]$, $g(1) = 0$, the function T defined in (3.7) is a continuous Archimedean t–norm. t–norms with additive generators (continuous or discontinuous) are necessarily Archimedean.

Note 3.40. A t–norm with an additive generator is continuous if and only if its additive generator is continuous.

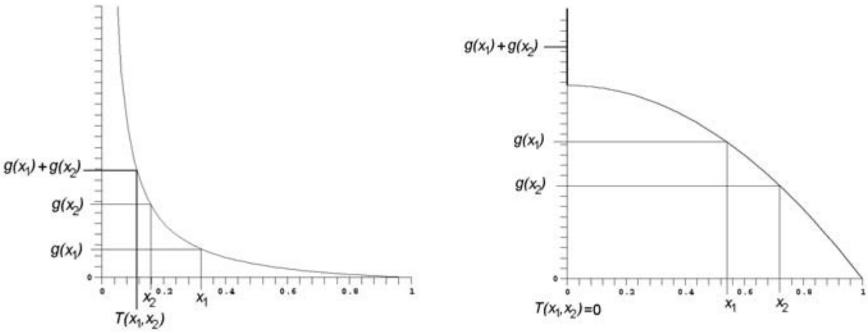


Fig. 3.4. Construction of strict (left) and nilpotent t–norms (right) using additive generators.

Example 3.41. Additive generators of the basic t–norms

1. If $g(t) = 1 - t$ we obtain the Łukasiewicz t–norm,
2. If $g(t) = -\log t$ we obtain the product t–norm.

Note 3.42. The minimum t–norm has no additive generator (it is not Archimedean).

Proposition 3.43. *An additive generator is defined up to a positive multiplicative constant, i.e., if $g(t)$ is an additive generator of T, then $cg(t), c > 0$ is also an additive generator of T.*

Thus we can have multiple additive generators of the same Archimedean t–norm T, for example $g(t) = -2\log(t) = -\log(t^2)$ and $g(t) = -\log(t)$ are both additive generators of the product t-norm.

Proposition 3.44. *If $g : [0,1] \to [0,\infty]$ is an additive generator of a continuous Archimedean t–norm T, then:*

- *T is strict if and only if $g(0) = \infty$;*
- *T is nilpotent if and only if $g(0) < \infty$.*

By using duality, we obtain analogous definitions and properties for t–conorms. Note though, that the additive generators of continuous Archimedean t–conorms are strictly increasing.

Proposition 3.45. *Let S be a continuous Archimedean t–conorm. Then it can be written as*

$$S(x, y) = h^{(-1)}(h(x) + h(y)), \tag{3.8}$$

where $h : [0,1] \to [0,\infty]$, $h(0) = 0$, is a continuous strictly increasing function, called an additive generator of S [7].

Note 3.46. The converse of Proposition 3.45 is also true: Given any continuous strictly increasing function $h : [0,1] \to [0,\infty]$, $h(0) = 0$, the function S defined in (3.8) is a continuous Archimedean t–conorm.

Due to the duality between t–norms and t–conorms, additive generators of t–conorms can be obtained from the additive generators of their dual t–norms.

Proposition 3.47. *Let T be a t–norm, S its dual t–conorm, and $g : [0,1] \to [0,\infty]$ an additive generator of T. The function $h : [0,1] \to [0,\infty]$ defined by $h(t) = g(1-t)$ is an additive generator of S.*

By using duality we also obtain the following results:

- An additive generator of a t–conorm is defined up to an arbitrary positive multiplier.
- A t–conorm with an additive generator is continuous if and only its additive generator is continuous.

[7] The expressions for pseudoinverse for continuous strictly increasing functions change to

$$h^{(-1)}(t) = \begin{cases} h^{-1}(t), & \text{if } t \in [0,a], \text{ where } a = h(1) \\ 1 & \text{otherwise.} \end{cases}$$

and $h^{(-1)}(t) = \sup\{z \in [0,1] | \, h(z) < t\}$, see also Section 3.4.6.

- A continuous Archimedean t–conorm with an additive generator h is strict if and only if $h(1) = \infty$.
- A continuous Archimedean t–conorm with an additive generator h is nilpotent if and only if $h(1) < \infty$.

Example 3.48. Additive generators of the basic t–conorms

1. If $h(t) = t$ we obtain the Łukasiewicz t–conorm,
2. If $h(t) = -\log(1 - t)$ we obtain the probabilistic sum t–conorm.

Note 3.49. The maximum t–conorm has no additive generators (it is not Archimedean).

From an additive generator $g : [0, 1] \to [0, \infty]$, of any t–norm it is possible to define its corresponding multiplicative generator in the form $\theta(t) = e^{-g(t)}$. The function $\theta : [0, 1] \to [0, 1]$, $\theta(1) = 1$ is continuous and strictly increasing.

Proposition 3.50. *Let T be a continuous Archimedean t–norm. Then it can be written as*

$$T(x, y) = \theta^{(-1)}(\theta(x) \cdot \theta(y)), \tag{3.9}$$

where $\theta : [0, 1] \to [0, 1]$, $\theta(1) = 1$ is a continuous strictly increasing function, called a ***multiplicative generator*** *of T.*

For t–conorms we have the analogous result, if h is an additive generator of S, then the function $\varphi(t) = e^{-h(t)}$ is a strictly decreasing function $\varphi : [0, 1] \to [0, 1]$, $\varphi(0) = 1$ called multiplicative generator of S.

Proposition 3.51. *Let S be a continuous Archimedean t–conorm. Then it can be written as*

$$S(x, y) = \varphi^{(-1)}(\varphi(x) \cdot \varphi(y)), \tag{3.10}$$

where $\varphi : [0, 1] \to [0, 1]$, $\varphi(0) = 1$ is a continuous strictly decreasing function, called a ***multiplicative generator*** *of S.*

Note 3.52. Any continuous strictly increasing function $\theta : [0, 1] \to [0, 1]$ with $\theta(1) = 1$ defines a continuous Archimedean t–norm by (3.9). Any continuous strictly decreasing function $\varphi : [0, 1] \to [0, 1]$ with $\varphi(0) = 1$ defines a continuous Archimedean t–conorm by (3.10).

3.4.6 Pseudo-inverses

In this section we study the definition of pseudo-inverses in greater detail, which allows us to define them for discontinuous functions.

Definition 3.53 (Pseudo-inverse of a monotone function). *Let $g : [a, b] \to [c, d]$ be a monotone function, where $[a, b], [c, d]$ are subintervals of the extended real line. Then the* ***pseudo–inverse*** *of g, $g^{(-1)} : [c, d] \to [a, b]$ is defined by*

$$g^{(-1)}(t) = \sup\{z \in [a, b] \mid (g(z) - t)(g(b) - g(a)) < 0\}.$$

From the above definition we have:

Corollary 3.54. *Let $g : [a, b] \to [c, d]$ be a monotone function, where $[a, b], [c, d]$ are subintervals of the extended real line.*

- *If $g(a) < g(b)$ we obtain the following formula*

$$g^{(-1)}(t) = \sup\{z \in [a, b] \mid g(z) < t\}$$

  for all $t \in [c, d]$.
- *If $g(a) > g(b)$ we obtain the following formula*

$$g^{(-1)}(t) = \sup\{z \in [a, b] \mid g(z) > t\}$$

  for all $t \in [c, d]$.
- *If $g(a) = g(b)$ for all $t \in [c, d]$ we have*

$$g^{(-1)}(t) = a.$$

The illustration of this is given in Fig. 3.5 which shows how to construct the graph of the pseudo–inverse $g^{(-1)}$ of the function $g : [0, 1] \to [0, a]$:

1 Draw vertical line segments at discontinuities of g.
2 Reflect the graph of g at the graph of the identity function on the extended real line (dashed line).
3 Remove any vertical line segments from the reflected graph except for their *lowest* points.

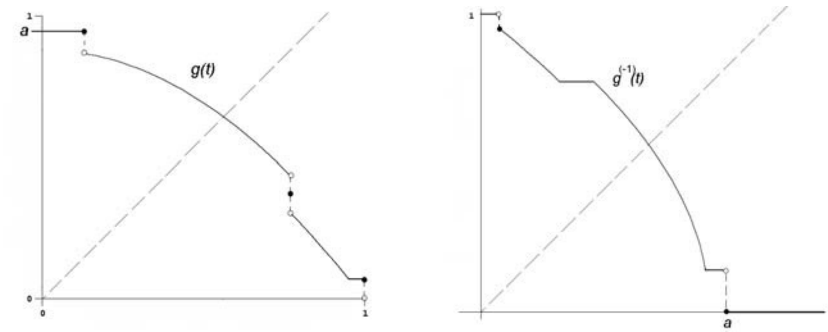


Fig. 3.5. Construction of the pseudo-inverse in the general case.

By using this more general definition of the pseudo-inverse, we can construct Archimedean t–norms which have discontinuous additive generators.

Example 3.55. The drastic product t–norm has a discontinuous additive generator

$$g(t) = \begin{cases} 2 - t, & \text{if } t \in [0, 1[, \\ 0, & \text{if } t = 1. \end{cases}$$

3.4.7 Isomorphic t–norms

Let us recall transformations of aggregation functions on p. 29, namely

$$g(\mathbf{x}) = \psi(f(\varphi(x_1), \varphi(x_2), \ldots, \varphi(x_n))),$$

where ψ, φ are univariate strictly increasing bijections. We saw that the resulting function g is an aggregation function, associative if f is associative and $\psi = \varphi^{-1}$. Let us now take an automorphism of the unit interval $\varphi : [0,1] \to [0,1]$. We remind that an automorphism of the interval $[0,1]$ is a continuous and strictly increasing function φ verifying $\varphi(0) = 0, \varphi(1) = 1$. An automorphism is obviously a bijection. Hence φ has the inverse φ^{-1}.

Example 3.56. Some examples of automorphisms $\varphi : [0,1] \to [0,1]$ are given below:

- $\varphi(t) = \frac{2t}{t+1}$.
- $\varphi(t) = t^\lambda$, $\lambda > 0$.
- $\varphi(t) = 1 - (1-t)^\lambda$, $\lambda > 0$.
- $\varphi(t) = \frac{\lambda^t - 1}{\lambda - 1}$, $\lambda > 0, \lambda \neq 1$.
- $\varphi(t) = \frac{t}{\lambda + (1-\lambda)t}$, $\lambda > 0$.
- $\varphi(t) = \frac{\log(1+\lambda t^\alpha)}{\log(1+\lambda)}$, $\lambda > -1, \alpha > 0$.

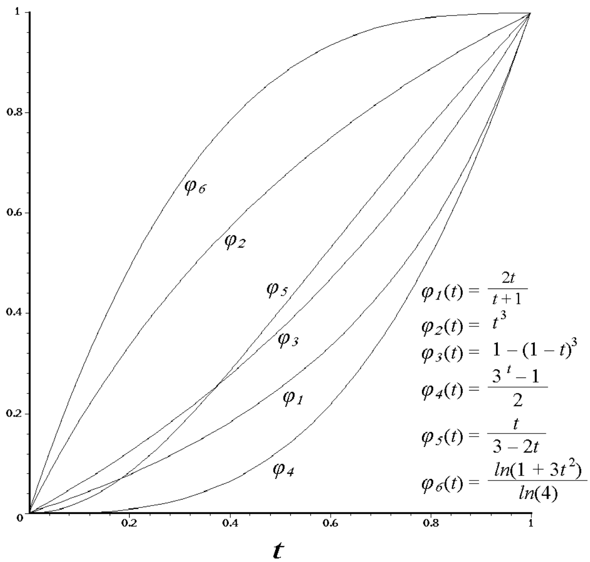


Fig. 3.6. Graphs of the automorphisms in Example 3.56 with a fixed parameter λ.

Note that if T is a t-norm and φ is an automorphism on the unit interval, then the function $T_\varphi : [0,1]^2 \to [0,1]$, defined as

$$T_\varphi(x, y) = \varphi^{-1}(T(\varphi(x), \varphi(y))) \tag{3.11}$$

is also a t-norm, which is said to be *isomorphic*[8] to T.

Proposition 3.57. *Let T be an Archimedean t–norm with an additive generator $g : [0, 1] \to [0, \infty]$, $g(1) = 0$, and φ an automorphism of $[0, 1]$.*

1. *The function*
$$T_\varphi(x, y) = \varphi^{-1}(T(\varphi(x), \varphi(y)))$$
is an Archimedean t–norm, isomorphic to T.
2. *$\tilde{g} = g \circ \varphi : [0, 1] \to [0, \infty]$ is an additive generator of the Archimedean t–norm T_φ.*
3. *T_φ is continuous if and only if T is continuous.*
4. *T_φ is strict if and only if T is strict.*
5. *T_φ is nilpotent if and only if T is nilpotent.*

One is not restricted to the automorphisms of the unit interval, strictly increasing bijections $\psi : [0, \infty] \to [0, \infty]$ can also be used to obtain t–norms isomorphic to a given t–norm T:

Proposition 3.58. *Let T be a continuous Archimedean t–norm with an additive generator $g : [0, 1] \to [0, \infty]$, and let ψ be a strictly increasing bijection $[0, \infty] \to [0, \infty]$.*

1. *The function $\hat{g} = \psi \circ g : [0, 1] \to [0, \infty]$ is an additive generator of a continuous Archimedean t–norm $\hat{T}_\psi$ isomorphic to T.*[9]
3. *$\hat{T}_\psi$ is strict if and only if T is strict.*
4. *$\hat{T}_\psi$ is nilpotent if and only if T is nilpotent.*

The next statement is a very strong result, which establishes that all continuous Archimedean t–norms are in essence isomorphic to the two prototypical examples: the product and the Łukasiewicz t–norms.

[8] In mathematics, the term *isomorphism* is usually applied to algebraic structures such as semigroups. A detailed discussion of isomorphisms of semigroups, and of t–norms and t–conorms as semigroup operations, is given in [142], pp. 37-38.

[9] Note that $\hat{T}_\psi$ is not related to T in (3.11) with $\psi = \varphi$. However, for any strictly increasing bijection ψ there exists an automorphism of $[0, 1]$ φ, such that T_φ is isomorphic to T as in (3.11), and $\hat{T}_\psi = T_\varphi$. For a strict t–norm take $\varphi = g^{-1} \circ \psi \circ g$. For a nilpotent t–norm we take $\varphi = g^{(-1)} \circ \tilde{\psi} \circ g$, where $\tilde{\psi}(t) = \psi(t) \frac{g(0)}{\psi(g(0))}$.

Proposition 3.59. *Let T be a continuous Archimedean t–norm.*

- *If T is strict, then it is isomorphic to the product t–norm T_P, i.e., there exists an automorphism of the unit interval φ such that $T_\varphi = T_P$.*
- *If T is nilpotent, then it is isomorphic to the Łukasiewicz t–norm T_L, i.e., there exists an automorphism of the unit interval φ such that $T_\varphi = T_L$.*

Let us now check what will happen with the other two basic t–norms, the drastic product and the minimum (we remind that neither of these is strict or nilpotent: the minimum is not Archimedean, and the drastic product is not continuous). It is easy to check that under any automorphism φ, $T_{min_\varphi} = T_{min}$ and $T_{D_\varphi} = T_D$, i.e., these two t–norms do not change under any automorphism.

This does not mean that all t–norms (or even all continuous t–norms) are isomorphic to just the four basic t–norms, there are many t–norms that are continuous but not Archimedean. We will see, however, in the next section, that all continuous t–norms are either isomorphic to T_P, T_L, or can be constructed from these two t–norms and the minimum (the ordinal sum construction).

By using automorphisms we can construct new families of t–norms based on an existing t–norm.

Example 3.60. Consider the Łukasiewicz t–norm T_L.

- Take $\varphi(t) = t^\lambda$ ($\lambda > 0$), since it is $\varphi^{-1}(t) = t^{1/\lambda}$, we get:
$$T_\varphi(x, y) = (\max(0, x^\lambda + y^\lambda - 1))^{1/\lambda}.$$
- Take $\varphi(t) = 1 - (1 - t)^\lambda$ ($\lambda > 0$), the inverse is $\varphi^{-1}(t) = 1 - (1 - t)^{1/\lambda}$, and therefore:
$$T_\varphi(x, y) = 1 - [\min(1, (1 - x)^\lambda + (1 - y)^\lambda)]^{1/\lambda}.$$

Example 3.61. Consider the product t–norm T_P.

- Take $\varphi(t) = 1 - (1 - t)^\lambda$ ($\lambda > 0$), then
$$T_\varphi(x, y) = 1 - [(1 - x)^\lambda + (1 - y)^\lambda - (1 - x)^\lambda (1 - y)^\lambda)]^{1/\lambda}.$$
- Take $\varphi(t) = \frac{2t}{t+1}$, since $\varphi^{-1}(t) = \frac{t}{2-t}$, we get:
$$T_\varphi(x, y) = \frac{2xy}{x + y + 1 - xy}.$$

Moreover, Propositions 3.57 and 3.58 present an opportunity to construct, based on any continuous Archimedean t–norm, interesting families of t–norms by modifying their additive generators.

Example 3.62. [142] Let T be a continuous Archimedean t–norm and $g : [0,1] \to [0,\infty]$ its additive generator.

- For each $\lambda \in]0,\infty[$ the function $\psi \circ g = g^\lambda : [0,1] \to [0,\infty]$ is also an additive generator of a continuous Archimedean t–norm which we will denote by $T^{(\lambda)}$. The family of these t–norms is strictly increasing with respect to parameter λ, and, curiously, adding the limit cases to this family of t–norms, i.e., $T^{(0)} = T_D$ and $T^{(\infty)} = \min$ we get well-known families of t–norms of Yager, Aczél–Alsina and Dombi (see Section 3.4.11) depending on the initial t-norm is T_L, T_P or T_0^H respectively.
- Let T^* be a strict t–norm with an additive generator $g^* : [0,1] \to [0,\infty]$. Then for each $\lambda \in]0,\infty]$, the function $g_{(g^*,\lambda)} : [0,1] \to [0,\infty]$ defined by

$$g_{(g^*,\lambda)}(t) = g((g^*)^{-1}(\lambda g^*(t)))$$

is an additive generator of a continuous Archimedean t–norm which we will denote by $T_{(T^*,\lambda)}$. For example, for $\lambda \in]0,\infty]$ we have $T_{P(T_0^H,\lambda)} = T_\lambda^H$, the Hamacher t–norms, see p. 152.

3.4.8 Comparison of continuous Archimedean t–norms

In Section 1.3.4 we defined standard pointwise comparison of aggregation functions. We have seen that the basic t–norms verify

$$T_D \leq T_L \leq T_P \leq \min .$$

However, not all t–norms are comparable, although many of them, especially those from parametric families, are. It is possible to find couples of incomparable strict and nilpotent t–norms.

The incomparability of two continuous Archimedean t–norms should be viewed based on the properties of their additive generators. This idea was introduced by Schweizer and Sklar in [220, 221]. We summarize their results.

Proposition 3.63. *Let T_1 and T_2 be two continuous Archimedean t–norms with additive generators $g_1, g_2 : [0,1] \to [0,\infty]$, respectively.*

(i) *The following are equivalent:*
  1. $T_1 \leq T_2$
  2. *The function $(g_1 \circ g_2^{-1}) : [0, g_2(0)] \to [0,\infty]$ is subadditive, i.e., for all $x, y \in [0, g_2(0)]$ with $x + y \in [0, g_2(0)]$ we have*

$$(g_1 \circ g_2^{-1})(x+y) \leq (g_1 \circ g_2^{-1})(x) + (g_1 \circ g_2^{-1})(y)$$

(ii) *If the function $s = (g_1 \circ g_2^{-1}) : [0, g_2(0)] \to [0,\infty]$ is concave* [10]*, then we get $T_1 \leq T_2$.*

[10] I.e., $s(at_1 + (1-a)t_2) \geq as(t_1) + (1-a)s(t_2)$, holds for all $t_1, t_2 \in Dom(s)$ and $0 \leq a \leq 1$.

(iii) If the function $f(t) = \frac{(g_1 \circ g_2^{-1})(t)}{t} :]0, g_2(0)] \to [0, \infty]$ *is non-increasing then we get* $T_1 \le T_2$.

Note 3.64. Each concave function $f : [0, a] \to [0, \infty]$ verifying $f(0) = 0$ is subadditive. However, in general, subadditivity of a function does not imply its concavity. For instance the following strictly increasing and continuous function $g : [0, \infty] \to [0, \infty]$ is subadditive but evidently not concave

$$g(t) = \begin{cases} 3t, & \text{if } t \in [0, 3] \ , \\ t + 7, & \text{if } t \in]3, 5], \\ 2t + 1 & \text{otherwise.} \end{cases}$$

3.4.9 Ordinal sums

We now consider an interesting and powerful construction, which allows one to build new t–norms/t–conorms from scaled versions of existing t–norms/t–conorms. It works as follows: consider the domain of a bivariate t–norm, the unit square $[0, 1]^2$, see Fig. 3.7. Take the diagonal of this square, and define smaller squares on that diagonal as shown (define them using the upper and lower bounds, e.g., bounds a_1, b_1 define $[a_1, b_1]^2$). Now define T on each of the squares $[a_i, b_i]^2$ as a scaled t–norm T_i (as in the following definition), and *minimum* everywhere else. The resulting function T defined in such pointwise manner will be itself a t–norm, called an ordinal sum.

Definition 3.65 (Ordinal sum). *Let* $(T_i)_{i=1,...,K}$ *be a family of t–norms and* $(]a_i, b_i[)_{i=1,...,K}$ *be a family of non–empty, pairwise disjoint open subintervals of* $[0, 1]$. *The function* $T : [0, 1]^2 \to [0, 1]$ *given by*

$$T(x, y) = \begin{cases} a_i + (b_i - a_i) T_i\left(\frac{x - a_i}{b_i - a_i}, \frac{y - a_i}{b_i - a_i}\right), & \text{if } (x, y) \in [a_i, b_i]^2 \\ \min(x, y) & \text{otherwise} \end{cases} \tag{3.12}$$

is a t–norm which is called the ordinal sum of the summands $< a_i, b_i, T_i >$, $i = 1, \dots, K$, *and denoted by* $T = (< a_i, b_i, T_i >)_{i=1,...,K}$.

Note 3.66. In fact one has to *prove* that the resulting ordinal sum is a t–norm, the key issue being associativity. This was done in [155] and we take this for granted.

Example 3.67.

1. Each t–norm T is a trivial ordinal sum with just one summand $< 0, 1, T >$, i.e, we have $T = (< 0, 1, T >)$.
2. The ordinal sum $T = (< 0.2, 0.5, T_P >, < 0.5, 0.8, T_L >)$ is given by

$$T(x, y) = \begin{cases} 0.2 + \frac{(x - 0.2)(y - 0.2)}{0.3}, & \text{if } (x, y) \in [0.2, 0.5]^2 \ , \\ 0.5 + \max(x + y - 1.3, 0), & \text{if } (x, y) \in [0.5, 0.8]^2 \ , \\ \min(x, y) & \text{otherwise.} \end{cases}$$

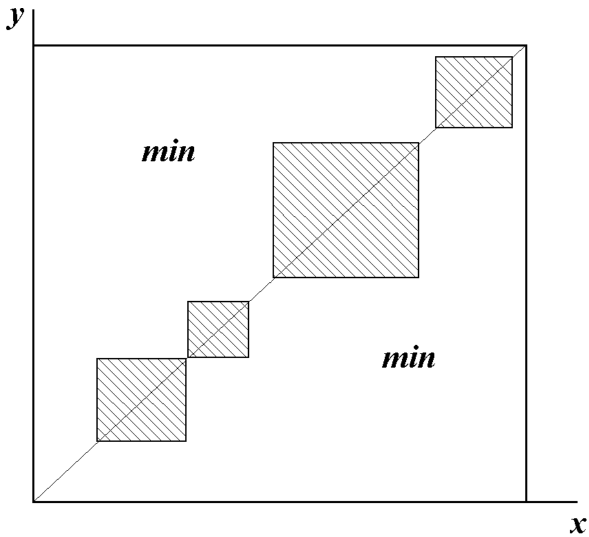


Fig. 3.7. Construction of ordinal sums: parts of the domain on which the summands are applied.

3. An ordinal sum of t–norms may have infinite summands. For example [142] $T = (< \frac{1}{2^{n+1}}, \frac{1}{2^n}, T_P >)_{n \in \mathbb{N}}$ is given by

$$T(x,y) = \begin{cases} \frac{1}{2^{n+1}} + 2^{n+1}(x - \frac{1}{2^{n+1}})(y - \frac{1}{2^{n+1}}), & \text{if } (x,y) \in [\frac{1}{2^{n+1}}, \frac{1}{2^n}]^2, \\ \min(x,y) & \text{otherwise.} \end{cases}$$

The 3D plots of the two mentioned ordinal sums are shown on Fig. 3.8.

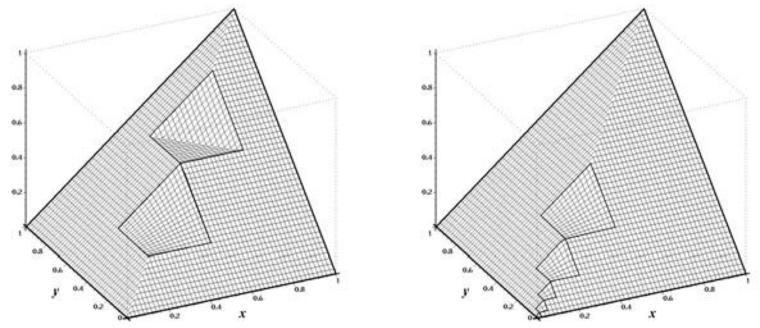


Fig. 3.8. 3D plots of the ordinal sums in Example 3.67 (with $n = 1, \dots, 4$ on the right).

For the case of t–conorms we have an analogous definition

Definition 3.68 (Ordinal sum of t–conorms). *Let $(S_i)_{i=1,\dots,K}$ be a family of t–conorms and $(]a_i, b_i[)_{i=1,\dots,K}$ be a family of non–empty, pairwise disjoint open subintervals of $[0,1]$. The function $S : [0,1]^2 \to [0,1]$ defined by*

$$S(x,y) = \begin{cases} a_i + (b_i - a_i) S_i\left(\frac{x-a_i}{b_i-a_i}, \frac{y-a_i}{b_i-a_i}\right), & \text{if } (x,y) \in [a_i,b_i]^2, \\ \max(x,y) & \text{otherwise} \end{cases} \tag{3.13}$$

is a t–conorm which is called the ordinal sum of summands $< a_i, b_i, S_i >, i = 1, \dots, K$, and denoted by $S = (< a_i, b_i, S_i >)_{i=1,\dots,K}$.

Note 3.69. Let $(< a_i, b_i, T_i >)_{i=1,\dots,K}$ be an ordinal sum of t–norms. Then the dual t–conorm is just an ordinal sum of t–conorms, i.e., $(< 1 - b_i, 1 - a_i, S_i >)_{i=1,\dots,K}$, where each t–conorm S_i is the dual of the t–norm T_i.

If each summand of an ordinal sum of t–norms is a continuous Archimedean t–norm, i.e., it has an additive generator, then the ordinal sum also has additive generators:

Proposition 3.70. *Let $(T_i)_{i=1,\dots,K}$ be a family of t–norms and assume that for each $i \in I = \{1, \dots, K\}$ the t–norm T_i has an additive generator $g_i : [0,1] \to [0,\infty]$. For each $i \in I$ define the function $h_i : [a_i, b_i] \to [0, \infty]$ by $h_i = g_i \circ \varphi_i$, where $\varphi_i : [a_i, b_i] \to [0, \infty]$ is given by $\varphi_i(t) = \frac{t-a_i}{b_i-a_i}$. Then for all $(x,y) \in [0,1]^2$*

$$T(x,y) = \begin{cases} h_i^{(-1)}(h_i(x) + h_i(y)), & \text{if } (x,y) \in [a_i,b_i]^2 \;, \\ \min(x,y) & \text{otherwise.} \end{cases}$$

We can see that an additive generator of each summand of the ordinal sum is defined on the corresponding subinterval $[a_i, b_i]$ by means of an isomorphism acting on an additive generator g_i. This is equivalent to re-scaling the generator. This allows one to define the ordinal sums of Archimedean t–norms using additive generators of the summands.

Example 3.71.

1. For the ordinal sum T in the Example 3.67(2) the functions $h_1 : [0.2, 0.5] \to [0, \infty], h_2 : [0.5, 0.8] \to [0, \infty]$ are given by, respectively,

$$\begin{aligned} h_1(t) &= -\log(\tfrac{10t-2}{3}), \\ h_2(t) &= \tfrac{8-10t}{3}. \end{aligned}$$

2. In the Example 3.67(3), for each $n \in \mathbf{N}$ the function $h_i : [\frac{1}{2^{n+1}}, \frac{1}{2^n}] \to [0, \infty]$ is given by

$$h_i(t) = -\log(2^{n+1}t - 1).$$

Note 3.72. The representation of a t–norm as an ordinal sum of t–norms is not unique, in general. For instance, we have for each subinterval $[a, b]$ of $[0, 1]$ $\min =< (0, 1, \min) >=< (a, b, \min) >$.

There are many uses of the ordinal sum construction. Firstly, it allows one to prove certain theoretical results, which would otherwise be hard to establish. Secondly, it gives a way to define t–norms/t–conorms with some desired properties and behavior. We shall use this construction in Section 3.7.

However one of the most important results based on this construction is the following.

Proposition 3.73 (Continuous t–norms classification). *A continuous t–norm T is either*

- *Isomorphic to the product t–norm T_P (hence T is strict);*
- *Isomorphic to the Łukasiewicz t–norm T_L (hence T is nilpotent);*
- *T is* minimum*;*
- *T is a non-trivial ordinal sum of continuous Archimedean t–norms.*

For t–conorms we obtain an analogous result by duality.

Proposition 3.74 (Continuous t–conorms classification). *A continuous t–conorm S is either*

- *Isomorphic to the dual product t–conorm S_P (hence S is strict);*
- *Isomorphic to the Łukasiewicz t–conorm S_L (hence S is nilpotent);*
- *S is* maximum*;*
- *S is a non-trivial ordinal sum of continuous Archimedean t–conorms.*

3.4.10 Approximation of continuous t–norms

Continuous Archimedean t–norms and t–conorms have a very nice and useful representation through their additive or multiplicative generators. Essentially, one has to deal with a continuous strictly monotone *univariate* function in order to generate a whole family of n-variate aggregation functions. Of course, this representation does not hold for all continuous t–norms (t–conorms), which can be non-Archimedean (in which case they are either *minimum* (*maximum* for t–conorms) or *ordinal sums* of Archimedean t–norms (t–conorms)).

Our next question is whether any continuous t–norm can be *approximated* sufficiently well with a continuous Archimedean t–norm, in which case we could apply the mechanism of additive generators to represent up to a certain precision all continuous t–norms. The answer to this question is positive. It was established in [136, 137], see also [142].

Proposition 3.75 (Approximation of a continuous t–norm). *Any continuous t–norm T can be approximated uniformly*[11] *with any desired precision ε by some continuous Archimedean t–norm T_A.*

Expressed in other words, the set of continuous Archimedean t–norms *is dense* in the set of all continuous t–norms. Of course the analogous result holds for t–conorms. Thus we can essentially substitute any continuous t–norm with a continuous Archimedean t–norm, in such a way that the values of both functions at any point $\mathbf{x} \in [0,1]^n$ do not differ by more than $\varepsilon \geq 0$, and ε can be made arbitrarily small. Specifically, when using t–norms (or t–conorms) as aggregation functions on a computer (which, of course, has finite precision), we can just use continuous Archimedean t–norms (t–conorms), as there will be no noticeable numerical difference between an Archimedean and any other continuous t–norm (t–conorm).

The second important result is related to the use of additive generators. We know that a continuous Archimedean t–norm can be represented by using its additive generators, which means that its value at any point (x, y) can be calculated using formula (3.7). The question is, if we take two additive generators that are close in some sense (e.g., pointwise), will the resulting t-norms be close as well? The answer to this question is also positive, as proved in [136], see also [142].

Proposition 3.76 (Convergence of additive generators). *Let $T_i, i = 1, 2, \ldots$ be a sequence of continuous Archimedean t–norms with additive generators $g_i, i = 1, 2, \ldots$, such that $g_i(0.5) = 1$, and let T be some continuous Archimedean t–norm with additive generator $g : g(0.5) = 1$. Then*

$$\lim_{i \to \infty} T_i = T$$

[11] Uniform approximation of f by $\tilde{f}$ means that $\max_{\mathbf{x} \in [0,1]^n} |f(\mathbf{x}) - \tilde{f}(\mathbf{x})| \leq \varepsilon$.

if and only if for each $t \in]0,1]$ *we have*

$$\lim_{i\to\infty} g_i(t) = g(t).$$

The convergence of the sequence of t–norms is pointwise and uniform. The condition $g_i(0.5) = 1$ is technical: since the additive generators are not defined uniquely, but up to an arbitrary positive multiplier, we need to fix somehow a single generator for a given t–norm, and we do it by using the mentioned condition.

Propositions 3.75 and 3.76 together imply that any continuous t–norm can be approximated (uniformly, and up to any desired accuracy) by approximating a univariate function – an additive generator. We will use this fact in Section 3.4.15, where we discuss constructions of t–norms and t–conorms based on empirically collected data.

3.4.11 Families of t–norms

In this section we want to provide the reader with a list of parameterized families of t–norms. We will consider the main families of t–norms and t–conorms: Schweizer-Sklar, Hamacher, Frank, Yager, Dombi, Aczel-Alsina, Mayor-Torrens and Weber-Sugeno t–norms and t–conorms, and closely follow [142]. Some of these families include the basic t–norms/t–conorms as the limiting cases. Because of associativity, we only need to provide the expressions in the bivariate case; n-variate formulae can be obtained by recursion.

We will also mention monotonicity of each of these families and their continuity with respect to the parameter.

Schweizer-Sklar

The family $(T_\lambda^{SS})_{\lambda\in[-\infty,\infty]}$ of Schweizer-Sklar t–norms is given by

$$T_\lambda^{SS}(x,y) = \begin{cases} \min(x,y), & \text{if } \lambda=-\infty,\\ T_P(x,y), & \text{if } \lambda=0,\\ T_D(x,y), & \text{if } \lambda=\infty,\\ \left(\max((x^\lambda+y^\lambda-1),0)\right)^{\frac{1}{\lambda}} & \text{otherwise.}\end{cases}$$

The family $(S_\lambda^{SS})_{\lambda\in[-\infty,\infty]}$ of Schweizer-Sklar t-conorms is given by

$$S_\lambda^{SS}(x,y) = \begin{cases} \max(x,y), & \text{if } \lambda=-\infty,\\ S_P(x,y), & \text{if } \lambda=0,\\ S_D(x,y), & \text{if } \lambda=\infty,\\ 1-\left(\max(((1-x)^\lambda+(1-y)^\lambda-1),0)\right)^{\frac{1}{\lambda}} & \text{otherwise.}\end{cases}$$

Limiting cases: $T_{-\infty}^{SS}=\min$,$T_0^{SS}=T_P$, $T_1^{SS}=T_L$, $T_\infty^{SS}=T_D$, $S_{-\infty}^{SS}=\max$, $S_0^{SS}=S_P$, $S_1^{SS}=S_L$, $S_\infty^{SS}=S_D$.

- For each $\lambda\in[-\infty,\infty]$ the t–norm T_λ^{SS} and the t–conorm S_λ^{SS} are dual to each other.
- T_λ^{SS} and S_λ^{SS} are continuous for all $\lambda\in[-\infty,\infty[$.
- T_λ^{SS} is Archimedean for $\lambda\in]-\infty,\infty]$.
- T_λ^{SS} is strict if and only if $\lambda\in]-\infty,0]$ and it is nilpotent if and only if $\lambda\in]0,\infty[$.

Additive generators g_λ^{SS},$h_\lambda^{SS}:[0,1]\to[0,\infty]$ of the continuous Archimedean t–norms and t–conorms are given by, respectively

$$g_\lambda^{SS}(t) = \begin{cases} -\log t, & \text{if } \lambda=0,\\ \frac{1-t^\lambda}{\lambda}, & \text{if } \lambda\in]-\infty,0[\bigcup]0,\infty[,\end{cases}$$

and

$$h_\lambda^{SS}(t) = \begin{cases} -\log(1-t), & \text{if } \lambda=0,\\ \frac{1-(1-t)^\lambda}{\lambda}, & \text{if } \lambda\in]-\infty,0[\bigcup]0,\infty[.\end{cases}$$

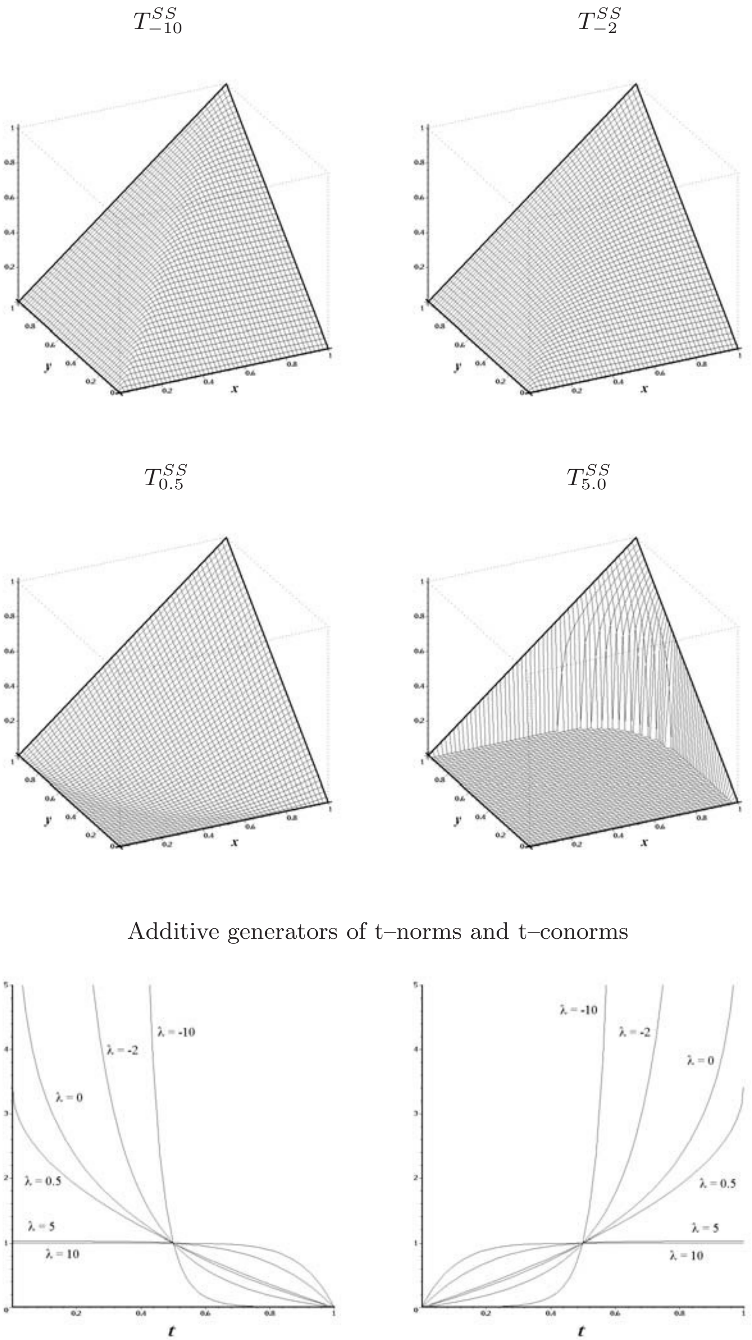


Fig. 3.9. 3D plots of some Schweizer-Sklar t–norms and their additive generators.

Hamacher

The family $(T^H_\lambda)_{\lambda\in[0,\infty]}$ of Hamacher t–norms is given by

$$T^H_\lambda(x,y)=\begin{cases} T_D(x,y), & \text{if } \lambda=\infty\ ,\\ 0, & \text{if } \lambda=x=y=0\ ,\\ \frac{xy}{\lambda+(1-\lambda)(x+y-xy)} & \text{otherwise}\ . \end{cases}$$

The family $(S^H_\lambda)_{\lambda\in[0,\infty]}$ of Hamacher t-conorms is given by

$$S^H_\lambda(x,y)=\begin{cases} S_D(x,y), & \text{if } \lambda=\infty,\\ 1, & \text{if } \lambda=0 \text{ and } x=y=1,\\ \frac{x+y-xy-(1-\lambda)xy}{1-(1-\lambda)xy} & \text{otherwise.} \end{cases}$$

Hamacher t-norms and t-conorms are the only strict t-norms (t-conorms) that can be expressed as *rational* functions.[12]

Limiting cases: $T^H_\infty=T_D$, $T^H_1=T_P$ and $S^H_\infty=S_D$, $S^H_1=S_P$. Moreover, T^H_0 and S^H_2 are given by

$$T^H_0(x,y)=\begin{cases} 0, & \text{if } x=y=0,\\ \frac{xy}{x+y-xy} & \text{otherwise} \end{cases}$$

$$S^H_2(x,y)=\frac{x+y}{1+xy},$$

are respectively called *Hamacher product* and the *Einstein sum.*

- For each $\lambda\in[0,\infty]$ the t–norm T^H_λ and the t–conorm S^H_λ are dual to each other.
- All Hamacher t–norms are Archimedean, and all Hamacher t–norms with the exception of T^H_∞ are strict.
- There are no nilpotent Hamacher t–norms.
- The family of Hamacher t–norms is strictly decreasing with λ and the family of Hamacher t–conorms is strictly increasing with λ.
- The family of Hamacher t–norms is continuous with respect to the parameter λ, i.e., $\forall\lambda_0\in[0,\infty]$ we have $\lim\limits_{\lambda\to\lambda_0} T^H_\lambda=T^H_{\lambda_0}$.

Additive generators g^H_λ , $h^H_\lambda:[0,1]\to[0,\infty]$ of the continuous Archimedean Hamacher t–norms and t–conorms are given by, respectively,

$$g^H_\lambda(t)=\begin{cases} \frac{1-t}{t}, & \text{if } \lambda=0,\\ \log\frac{(1-\lambda)t+\lambda}{t}, & \text{if } \lambda\in]0,\infty[, \end{cases}$$

and

$$h^H_\lambda(t)=\begin{cases} \frac{t}{1-t}, & \text{if } \lambda=0,\\ \log\frac{(1-\lambda)(1-t)+\lambda}{1-t}, & \text{if } \lambda\in]0,\infty[. \end{cases}$$

---

[12] A rational function is expressed as a ratio of two polynomials.

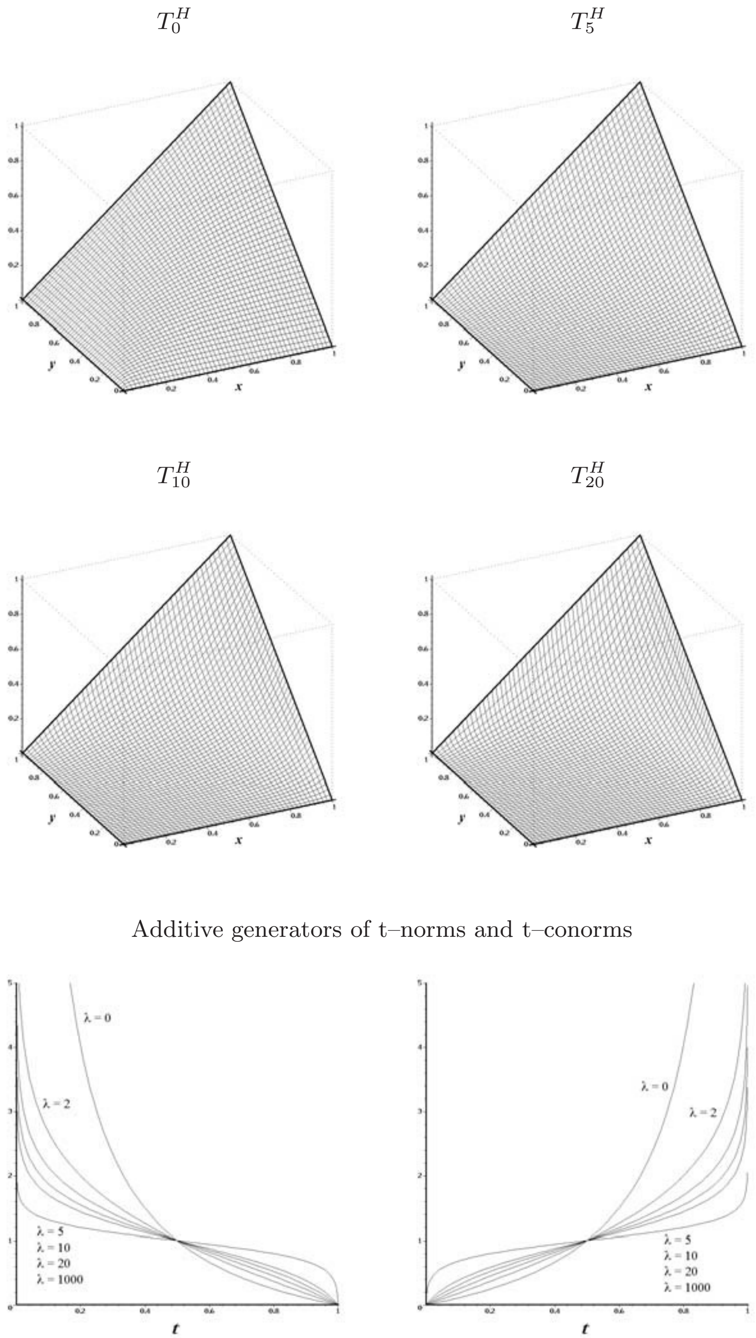


Fig. 3.10. 3D plots of some Hamacher t–norms and their additive generators

Frank t–norms

The origin of this family comes from the solutions of the following functional equation $G(x,y)+F(x,y)=x+y$ where F and G are associative functions and F satisfies $F(x,1)=F(1,x)=x$ and $F(x,0)=F(0,x)$. Frank shows that F has to be an ordinal sum of the following family of t–norms.

The family $(T_\lambda^F)_{\lambda\in[0,\infty]}$ of Frank t–norms is given by

$$T_\lambda^F(x,y)=\begin{cases}\min(x,y), & \text{if } \lambda=0\ ,\\ T_P(x,y), & \text{if } \lambda=1\ ,\\ T_L(x,y), & \text{if } \lambda=\infty\ ,\\ \log_\lambda(1+\frac{(\lambda^x-1)(\lambda^y-1)}{\lambda-1}) & \text{otherwise}\ .\end{cases}$$

The family $(S_\lambda^F)_{\lambda\in[0,\infty]}$ of Frank t–conorms is given by

$$S_\lambda^F(x,y)=\begin{cases}\max(x,y), & \text{if } \lambda=0\ ,\\ S_P(x,y), & \text{if } \lambda=1\ ,\\ S_L(x,y), & \text{if } \lambda=\infty,\\ 1-\log_\lambda(1+\frac{(\lambda^{1-x}-1)(\lambda^{1-y}-1)}{\lambda-1}) & \text{otherwise}\ .\end{cases}$$

Limiting cases: $T_0^F=\min$, $T_1^F=T_P$, $T_\infty^F=T_L$, $S_0^F=\max$, $S_1^F=S_P$, $S_\infty^F=S_L$.

- For each $\lambda\in[0,\infty]$ the t–norm T_λ^F and the t–conorm S_λ^F are dual to each other.
- T_λ^F and S_λ^F are continuous for all $\lambda\in[0,\infty]$.
- T_λ^F is Archimedean if and only if $\lambda\in]0,\infty]$.
- T_λ^F is strict if and only if $\lambda\in]0,\infty[$ and the unique nilpotent Frank t–norm is T_∞^F.

Additive generators $g_\lambda^F,h_\lambda^F:[0,1]\to[0,\infty]$ of the continuous Archimedean Frank t–norms and t–conorms are given by, respectively

$$g_\lambda^F(t)=\begin{cases}-\log t, & \text{if } \lambda=1,\\ 1-t, & \text{if } \lambda=\infty,\\ \log(\frac{\lambda-1}{\lambda^t-1}), & \text{if } \lambda\in]0,1[\bigcup]1,\infty[,\end{cases}$$

and

$$h_\lambda^F(t)=\begin{cases}-\log(1-t), & \text{if } \lambda=1,\\ t, & \text{if } \lambda=\infty\ ,\\ \log(\frac{\lambda-1}{\lambda^{1-t}-1}), & \text{if } \lambda\in]0,1[\bigcup]1,\infty[\ .\end{cases}$$

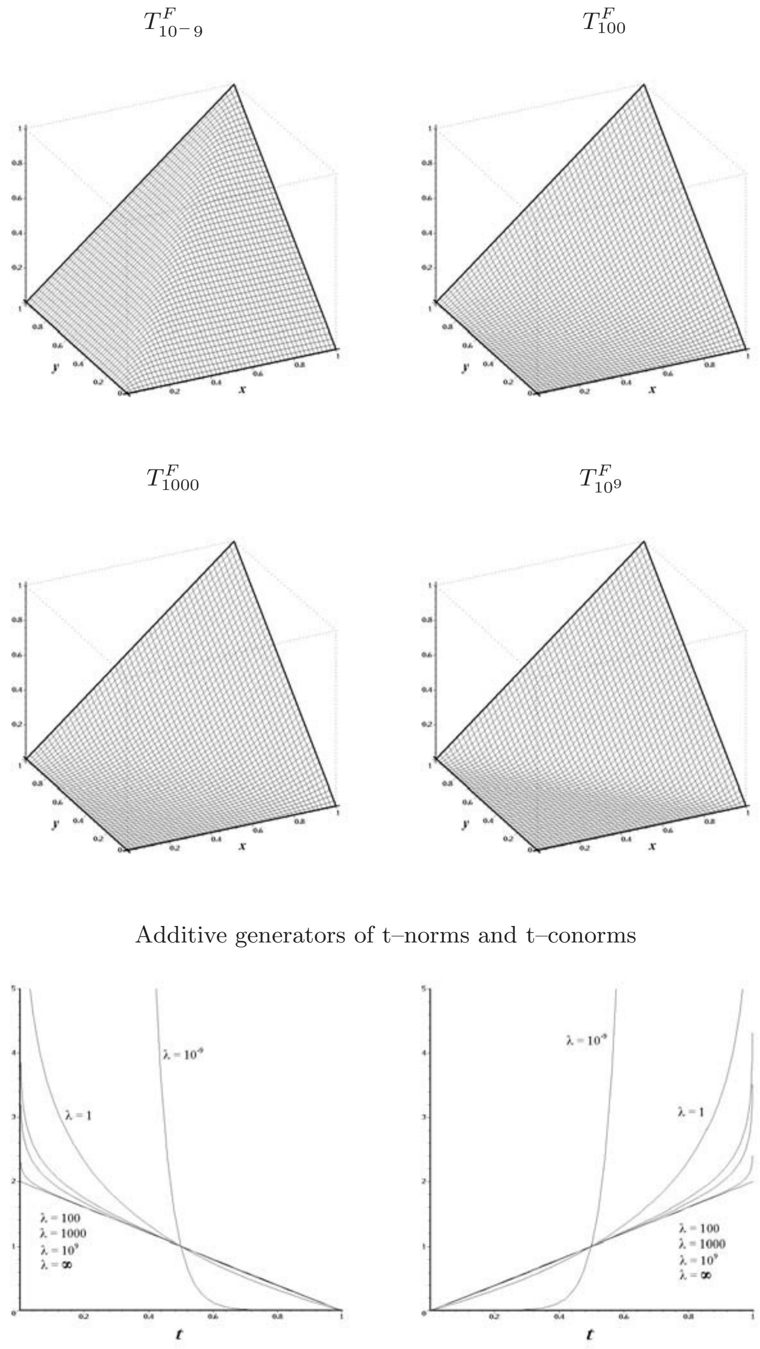


Fig. 3.11. 3D plots of some Frank t–norms and their additive generators.

Yager t–norms

This family was introduced by Yager in 1980 [262], his idea was to measure the strength of the logical AND by means of the parameter λ. Thus $\lambda = 0$ expresses the smallest AND, and $\lambda = \infty$ expresses the largest AND.

The family $(T^Y_\lambda)_{\lambda \in [0,\infty]}$ of Yager t–norms is given by

$$T^Y_\lambda(x,y) = \begin{cases} T_D(x,y), & \text{if } \lambda = 0 \ , \\ \min(x,y), & \text{if } \lambda = \infty \ , \\ \max(1 - ((1-x)^\lambda + (1-y)^\lambda)^{\frac{1}{\lambda}}, 0) & \text{otherwise} \ . \end{cases}$$

The family $(S^Y_\lambda)_{\lambda \in [0,\infty]}$ of Yager t–conorms is given by

$$S^Y_\lambda(x,y) = \begin{cases} S_D(x,y), & \text{if } \lambda = 0 \ , \\ \max(x,y), & \text{if } \lambda = \infty \ , \\ \min((x^\lambda + y^\lambda)^{\frac{1}{\lambda}}, 1) & \text{otherwise} \ . \end{cases}$$

Limiting cases: $T^Y_0 = T_D$, $T^Y_1 = T_L$, $T^Y_\infty = \min$, $S^Y_0 = S_D$, $S^Y_1 = S_L$, $S^Y_\infty = \max$.

- For each $\lambda \in [0,\infty]$ the t–norm T^Y_λ and the t–conorm S^Y_λ are dual to each other.
- T^Y_λ,S^Y_λ are continuous for all $\lambda \in]0,\infty]$.
- T^Y_λ, S^Y_λ are Archimedean if and only if $\lambda \in [0,\infty[$.
- T^Y_λ,S^Y_λ are nilpotent if and only if $\lambda \in]0,\infty[$, and none of the Yager t–norms are strict.

Additive generators $g^Y_\lambda, h^Y_\lambda : [0,1] \to [0,\infty]$ of the continuous Archimedean Yager t–norms and t–conorms are given by, respectively

$$g^Y_\lambda(t) = (1-t)^\lambda,$$

and

$$h^Y_\lambda(t) = t^\lambda.$$

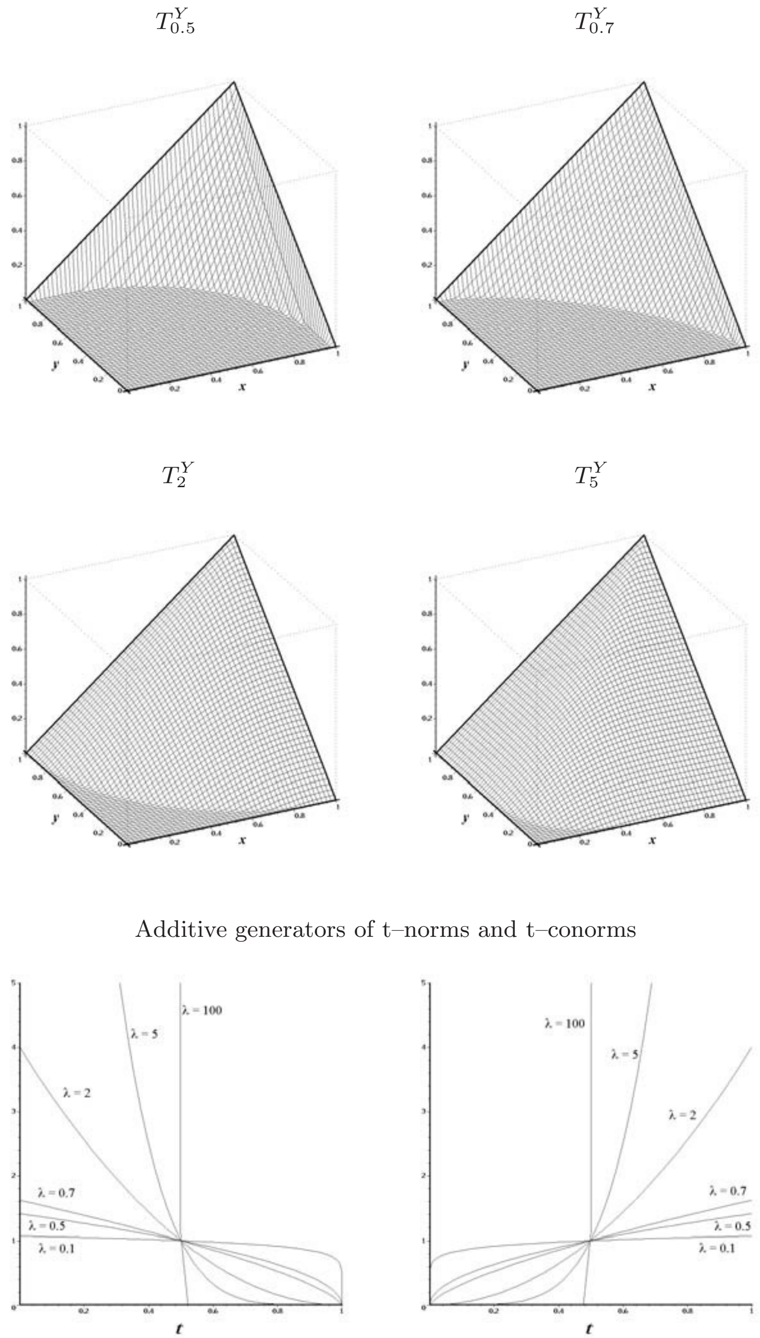


Fig. 3.12. 3D plots of some Yager t–norms and their additive generators.

Dombi t–norms

The following family of t–norms was introduced by Dombi in 1982 [79].

The family $(T_\lambda^D)_{\lambda\in[0,\infty]}$ of Dombi t–norms is given by

$$T_\lambda^D(x,y) = \begin{cases} T_D(x,y), & \text{if } \lambda = 0, \\ \min(x,y), & \text{if } \lambda = \infty, \\ \frac{1}{1+((\frac{1-x}{x})^\lambda + (\frac{1-y}{y})^\lambda)^{\frac{1}{\lambda}}} & \text{otherwise.} \end{cases}$$

The family $(S_\lambda^D)_{\lambda\in[0,\infty]}$ of Dombi t–conorms is given by

$$S_\lambda^D(x,y) = \begin{cases} S_D(x,y), & \text{if } \lambda = 0\ , \\ \max(x,y), & \text{if } \lambda = \infty\ , \\ 1 - \frac{1}{1+((\frac{x}{1-x})^\lambda + (\frac{y}{1-y})^\lambda)^{\frac{1}{\lambda}}} & \text{otherwise.} \end{cases}$$

Limiting cases: $T_0^D = T_D$, $T_1^D = T_0^H$, $T_\infty^D = \min$, $S_0^D = S_D$, $S_1^D = S_0^H$, $S_\infty^D = \max$.

- For each $\lambda \in [0,\infty]$ the t–norm T_λ^D and the t–conorm S_λ^D are dual to each other.
- T_λ^D are continuous for all $\lambda \in]0,\infty]$.
- T_λ^D is Archimedean if and only if $\lambda \in [0,\infty[$.
- T_λ^D is strict if and only if $\lambda \in]0,\infty[$, and none of the Dombi t–norms are nilpotent.
- The family of Dombi t–norms is strictly increasing and the family of Dombi t–conorms is strictly decreasing with parameter λ.

Additive generators $g_\lambda^D, h_\lambda^D : [0,1] \to [0,\infty]$ of the strict Dombi t–norms and t–conorms are given by, respectively

$$g_\lambda^D(t) = \left(\frac{1-t}{t}\right)^\lambda,$$

and

$$h_\lambda^D(t) = \left(\frac{t}{1-t}\right)^\lambda.$$

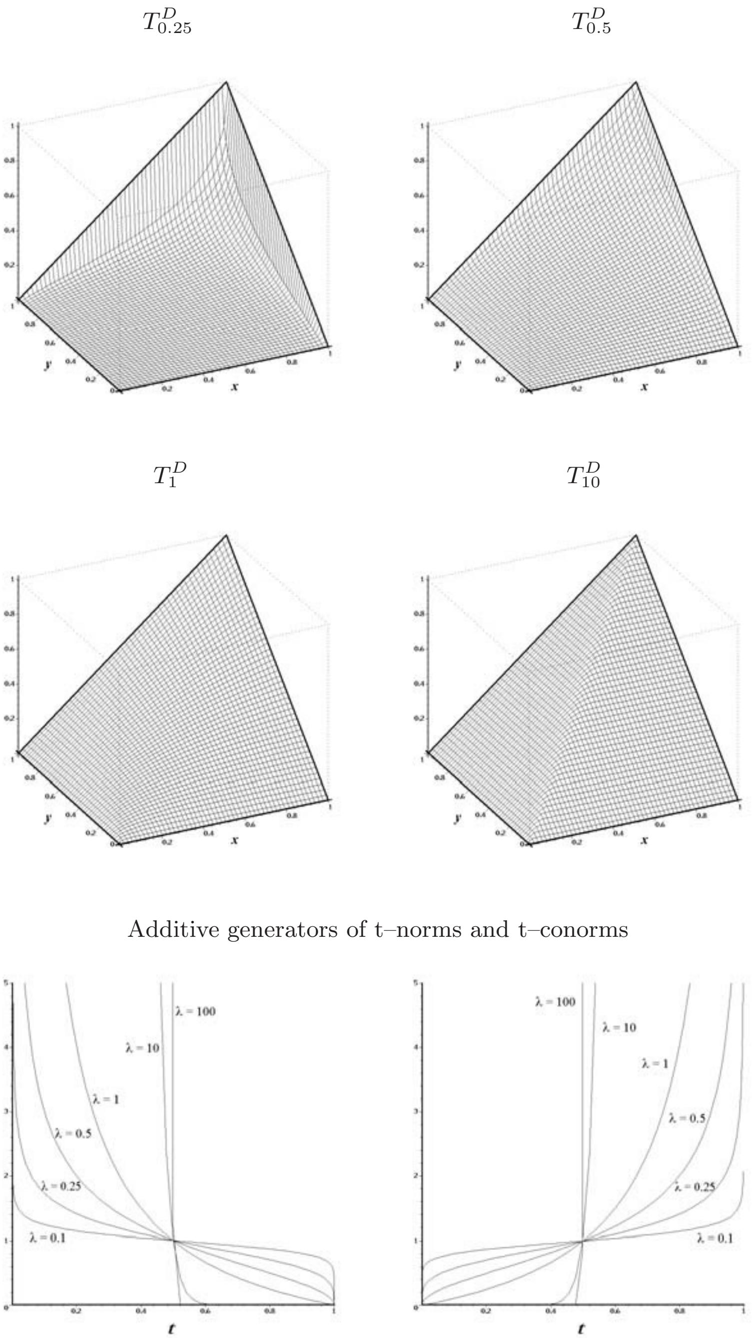


Fig. 3.13. 3D plots of some Dombi t–norms and their additive generators.

Aczél–Alsina t–norms

This family of t–norms was introduced by Aczél–Alsina in 1984 [2] in the context of functional equations.

The family $(T_\lambda^{AA})_{\lambda\in[0,\infty]}$ of Aczél–Alsina t–norms is given by

$$T_\lambda^{AA}(x,y) = \begin{cases} T_D(x,y), & \text{if } \lambda = 0 \ , \\ \min(x,y), & \text{if } \lambda = \infty \ , \\ e^{-((-\log x)^\lambda + (-\log y)^\lambda)^{\frac{1}{\lambda}}} & \text{otherwise} \ . \end{cases}$$

The family $(S_\lambda^{AA})_{\lambda\in[0,\infty]}$ of Aczél–Alsina t–conorms is given by

$$S_\lambda^{AA}(x,y) = \begin{cases} S_D(x,y), & \text{if } \lambda = 0 \ , \\ \max(x,y), & \text{if } \lambda = \infty \ , \\ 1 - e^{-((-\log(1-x))^\lambda + (-\log(1-y))^\lambda)^{\frac{1}{\lambda}}} & \text{otherwise} \ . \end{cases}$$

Limiting cases: $T_0^{AA} = T_D$, $T_1^{AA} = T_P$, $T_\infty^{AA} = \min$, $S_0^{AA} = S_D$, $S_1^{AA} = S_P$, $S_\infty^{AA} = \max$.

- For each $\lambda \in [0,\infty]$ the t–norm T_λ^{AA} and the t–conorm S_λ^{AA} are dual to each other.
- T_λ^{AA} are continuous for all $\lambda \in]0,\infty]$, T_0^{AA} is an exception.
- T_λ^{AA} is Archimedean if and only if $\lambda \in [0,\infty[$.
- T_λ^{AA} is strict if and only if $\lambda \in]0,\infty[$ and there are no nilpotent Aczél–Alsina t–norms.
- The family of Aczél–Alsina t–norms is strictly increasing and the family of Aczél–Alsina t–conorms is strictly decreasing.

Additive generators $g_\lambda^{AA}, h_\lambda^{AA} : [0,1] \to [0,\infty]$ of the strict Aczél–Alsina t–norms and t–conorms are given by, respectively,

$$g_\lambda^{AA}(t) = (-\log t)^\lambda,$$

and

$$h_\lambda^{AA}(t) = (-\log(1-t))^\lambda.$$

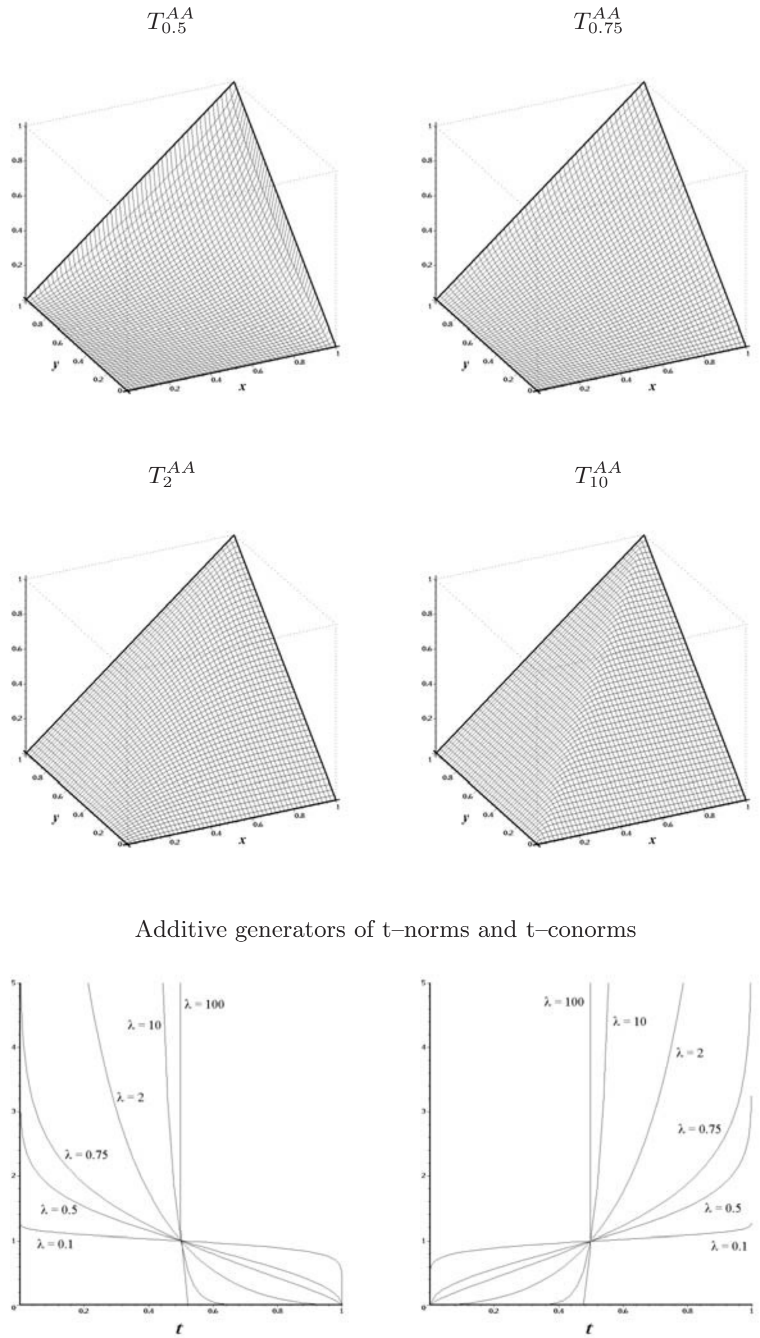


Fig. 3.14. 3D plots of some Aczél–Alsina t–norms and their additive generators.

Sugeno and Weber t–norms

In Weber 1983 [254] the use of some special t–norms and t–conorms was proposed to model the intersection and the union of fuzzy sets, respectively. This family of t–conorms has been considered previously in the context of λ–fuzzy measures in [230].

The family $(T_\lambda^{SW})_{\lambda\in[-1,\infty]}$ of Sugeno–Weber t–norms is given by

$$T_\lambda^{SW}(x,y) = \begin{cases} T_D(x,y), & \text{if } \lambda = -1 \ , \\ T_P(x,y), & \text{if } \lambda = \infty \ , \\ \max(\frac{x+y-1+\lambda xy}{1+\lambda}, 0) & \text{otherwise.} \end{cases}$$

The family $(S_\lambda^{SW})_{\lambda\in[-1,\infty]}$ of Sugeno–Weber t–conorms is given by

$$S_\lambda^{SW}(x,y) = \begin{cases} S_P(x,y), & \text{if } \lambda = -1 \ , \\ S_D(x,y), & \text{if } \lambda = \infty \ , \\ \min(x+y+\lambda xy, 1) & \text{otherwise.} \end{cases}$$

Limiting cases: $T_{-1}^{SW} = T_D$, $T_0^{SW} = T_L$, $T_\infty^{SW} = T_P$, $S_{-1}^{SW} = S_P$, $S_0^{SW} = S_L$, $S_\infty^{SW} = S_D$.

- For $\lambda, \mu \in]-1,\infty[$ the t–norm T_λ^{SW} and the t–conorm S_μ^{SW} are dual to each other if and only if $\mu = -\frac{\lambda}{1+\lambda}$. The following pairs $(T_{-1}^{SW}, S_\infty^{SW})$ and $(T_\infty^{SW}, S_{-1}^{SW})$ are also pairs of dual t–norms and t–conorms.
- All Sugeno–Weber t–norms with the exception of T_{-1}^{SW} are continuous.
- Each T_λ^{SW} is Archimedean, and it is nilpotent if and only if $\lambda \in]-1,\infty[$.
- The unique strict Sugeno–Weber t–norm is T_∞^{SW}.

Additive generators $g_\lambda^{SW}, h_\lambda^{SW}$: $[0,1] \to [0,\infty]$ of the continuous Archimedean t–norms and t–conorms of this family are given by, respectively

$$g_\lambda^{SW}(t) = \begin{cases} 1-t, & \text{if } \lambda = 0, \\ -\log t, & \text{if } \lambda = \infty, \\ 1 - \frac{\log(1+\lambda t)}{\log(1+\lambda)}, & \text{if } \lambda \in]-1,0[\bigcup]0,\infty[, \end{cases}$$

and

$$h_\lambda^{SW}(t) = \begin{cases} t, & \text{if } \lambda = 0, \\ -\log(1-t), & \text{if } \lambda = -1, \\ \frac{\log(1+\lambda t)}{\log(1+\lambda)}, & \text{if } \lambda \in]-1,0[\bigcup]0,\infty[. \end{cases}$$

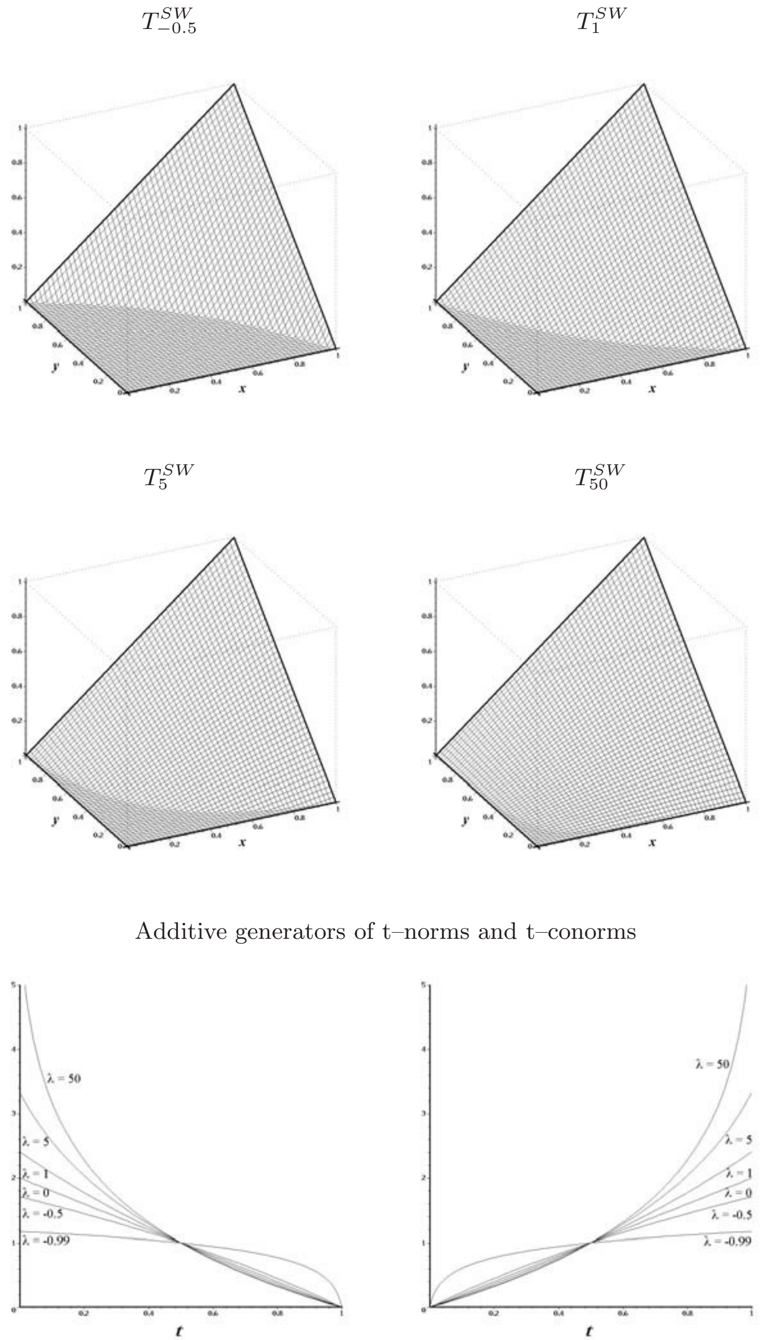


Fig. 3.15. 3D plots of some Sugeno and Weber t–norms and their additive generators.

Mayor and Torrens t–norms

The t–norms of this family are the only continuous t–norms that satisfy for all $x, y \in [0,1]$ the following equation

$$T(x,y) = \max(T(\max(x,y), \max(x,y)) - |x-y|, 0).$$

The family $(T_\lambda^{MT})_{\lambda\in[0,1]}$ of Mayor–Torrens t–norms is given by

$$T_\lambda^{MT}(x,y) = \begin{cases} \max(x+y-\lambda, 0), & \text{if } \lambda \in]0,1] \text{ and } (x,y) \in [0,\lambda]^2 \\ \min(x,y) & \text{otherwise.} \end{cases},$$

The family $(S_\lambda^{MT})_{\lambda\in[0,1]}$ of Mayor–Torrens t–conorms is given by

$$S_\lambda^{MT}(x,y) = \begin{cases} \min(x+y+\lambda-1, 1), & \text{if } \lambda \in]0,1] \text{ and } (x,y) \in [1-\lambda,1]^2 \\ \max(x,y) & \text{otherwise.} \end{cases},$$

Limiting cases: $T_0^{MT} = \min$, $S_0^{MT} = \max$,
$T_1^{MT} = T_L$ and $S_1^{MT} = S_L$.

- For each $\lambda \in [0,1]$ the t–norm T_λ^{MT} and the t–conorm S_λ^{MT} are dual to each other.
- The Mayor–Torrens t–norms and t–conorms are ordinal sums with one summand, respectively, i.e., $T_\lambda^{MT} = (< 0, \lambda, T_L >)$ and $S_\lambda^{MT} = (< 1-\lambda, 1, S_L >)$.
- Each T_λ^{MT} is a continuous t–norm.
- T_1^{MT} is a unique Archimedean and nilpotent t–norm. There are no Mayor–Torrens strict t–norms.
- The family of Mayor–Torrens t–norms is strictly decreasing with λ and the family of Mayor–Torrens t–conorms strictly increasing.
- The family of Mayor–Torrens t–norms is continuous with respect to the parameter λ, i.e., for all $\lambda_0 \in [0,1]$ we have $\lim_{\lambda\to\lambda_0} T_\lambda^{MT} = T_{\lambda_0}^{MT}$.

Additive generators of the nilpotent Mayor–Torrens t–norm T_1^{MT} and t–conorm S_1^{MT} are given by, respectively,

$$g_1^{MT}(t) = 1 - t,$$

and

$$h_1^{MT}(t) = t.$$

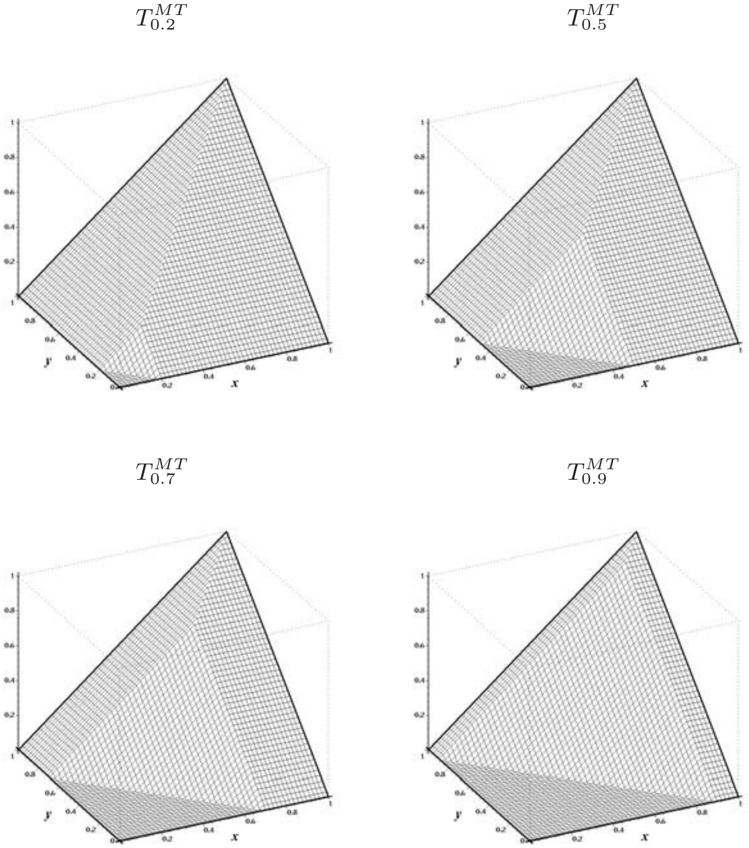


Fig. 3.16. 3D plots of some Mayor–Torrens t–norms.

Family	Continuous	Archimedean	Strict	Nilpotent	Additive generator	Inverse
Schweizer–Sklar $(T_\lambda^{SS})_{\lambda\in[-\infty,\infty]}$	$\lambda \in [-\infty,\infty[$	$\lambda \in]-\infty,\infty]$	$\lambda \in]-\infty,0]$	$\lambda \in]0,\infty[$	$g_\lambda^{SS} = \begin{cases} -\log t, & \text{if } \lambda = 0 \\ \frac{1-t^\lambda}{\lambda}, & \text{if } \lambda \in]-\infty,0[\bigcup]0,\infty[\end{cases}$	$(g_\lambda^{SS})^{-1} = \begin{cases} e^{-t}, & \text{if } \lambda = 0 \\ (-t\lambda+1)^{\frac{1}{\lambda}}, & \text{if } \lambda \in]-\infty,0[\bigcup]0,\infty[\end{cases}$
Hamacher $(T_\lambda^{H})_{\lambda\in[0,\infty]}$	$\lambda \in [0,\infty[$	all	$\lambda \in [0,\infty[$	none	$g_\lambda^{H} = \begin{cases} \frac{1-t}{t}, & \text{if } \lambda = 0 \\ \log(\frac{\lambda+(1-\lambda)t}{t}), & \text{if } \lambda \in]0,\infty[\end{cases}$	$(g_\lambda^{H})_\lambda^{-1} = \begin{cases} \frac{1}{1+t}, & \text{if } \lambda = 0 \\ \frac{\lambda}{e^t+\lambda-1}, & \text{if } \lambda \in]0,\infty[\end{cases}$
Frank $(T_\lambda^{F})_{\lambda\in[0,\infty]}$	all	$\lambda \in]0,\infty]$	$\lambda \in]0,\infty[$	$\lambda = \infty$	$g_\lambda^{F} = \begin{cases} -\log t, & \text{if } \lambda = 1 \\ 1-t, & \text{if } \lambda = \infty \\ \log(\frac{\lambda-1}{\lambda^t-1}), & \text{if } \lambda \in]0,1[\bigcup]1,\infty[\end{cases}$	$(g_\lambda^{F})^{-1} = \begin{cases} e^{-t} & \text{if } \lambda = 1 \\ 1-t, & \text{if } \lambda = \infty \\ \frac{\log(\frac{\lambda-1+e^t}{e^t})}{\log\lambda}, & \text{if } \lambda \in]0,1[\bigcup]1,\infty[\end{cases}$
Yager $(T_\lambda^{Y})_{\lambda\in[0,\infty]}$	$\lambda \in]0,\infty]$	$\lambda \in [0,\infty[$	none	$\lambda \in]0,\infty[$	$g_\lambda^{Y} = (1-t)^\lambda$	$(g_\lambda^{Y})^{-1} = 1 - t^{\frac{1}{\lambda}}$
Aczél–Alsina $(T_\lambda^{AA})_{\lambda\in[0,\infty]}$	$\lambda \in]0,\infty]$	$\lambda \in [0,\infty[$	$\lambda \in]0,\infty[$	none	$g_\lambda^{AA} = (-\log t)^\lambda$	$(g_\lambda^{AA})^{-1} = e^{-t^{\frac{1}{\lambda}}}$
Dombi $(T_\lambda^{D})_{\lambda\in[0,\infty]}$	$\lambda \in]0,\infty]$	$\lambda \in [0,\infty[$	$\lambda \in]0,\infty[$	none	$g_\lambda^{D} = (\frac{1-t}{t})^\lambda$	$(g_\lambda^{D})^{-1} = (\frac{1}{1+t^{\frac{1}{\lambda}}})$
Sugeno–Weber $(T_\lambda^{SW})_{\lambda\in[-1,\infty]}$	$\lambda \in]-1,\infty]$	all	$\lambda = \infty$	$\lambda \in]-1,\infty[$	$g_\lambda^{SW} = \begin{cases} 1-t, & \text{if } \lambda = 0 \\ -\log t, & \text{if } \lambda = \infty \\ 1 - \frac{\log(1+\lambda t)}{\log(1+\lambda)} & \text{otherwise} \end{cases}$	$(g_\lambda^{SW})^{-1} = \begin{cases} 1-t & \text{if } \lambda = 0 \\ e^{-t}, & \text{if } \lambda = \infty \\ \frac{(1+\lambda t)^{1-t})-1}{\lambda} & \text{otherwise} \end{cases}$
Mayor–Torrens $(T_\lambda^{MT})_{\lambda\in[0,1]}$	all	$\lambda = 1$	none	$\lambda = 1$	$g_1^{MT} = 1 - t$	$(g_1^{MT})^{-1} = 1 - t$

Table 3.1. Main families of t–norms.

3.4.12 Lipschitz–continuity

Recently the class of k–Lipschitz t-norms, whenever $k > 1$, has been characterized (see [187]). Note that 1-Lipschitz t–norms are copulas, see Section 3.5.

Definition 3.77 (k-Lipschitz t–norm). *Let $T : [0;1]^2 \to [0;1]$ be a t-norm and let $k \in [1, \infty[$ be a constant. Then T is k–Lipschitz if*

$$| T(x_1, y_1) - T(x_2, y_2) | \leq k(| x_1 - x_2 | + | y_1 - y_2 |)$$

for all $x_1, x_2, y_1, y_2 \in [0, 1]$.

Note 3.78. In other words, a k-Lipschitz bivariate t–norm has the Lipschitz constant k in $|| \cdot ||_1$ norm, see Definition 1.58. Of course $k \geq 1$, because of the condition $T(t, 1) = t$.

The k–Lipschitz property implies the continuity of the t–norm. Recall that a continuous t-norm can be represented by means of an ordinal sum of continuous Archimedean t–norms, and that a continuous Archimedean t–norm can be represented by means of a continuous additive generator [142, 155] (see Section 3.4.5). Characterization of all k–Lipschitz t-norms can be reduced to the problem of characterization of all Archimedean k-Lipschitz t-norms.

Note 3.79. It is easy to see that if a t–norm T is k–Lipschitz, it is also m–Lipschitz for any $m \in [k, \infty]$. The 1–Lipschitz t–norms are exactly those t–norms that are also copulas [221] (Section 3.5). A strictly decreasing continuous function $g : [0, 1] \to [0, \infty]$ with $g(1) = 0$ is an additive generator of a 1–Lipschitz Archimedean t-norm if and only if g is convex.

Definition 3.80 (k-convex function). *Let $g : [0, 1] \to [0, \infty]$ be a strictly monotone function and let $k \in]0, \infty[$ be a real constant. Then g will be called k–convex if*

$$g(x + k\varepsilon) - g(x) \leq g(y + \varepsilon) - g(y)$$

holds for all $x \in [0, 1[, y \in]0, 1[$; with $x \leq y$ and $\varepsilon \in]0, \min(1 - y, \frac{1-x}{k})]$.

Note 3.81. If $k = 1$ the function g is convex.

Note 3.82. If a strictly monotone function is k–convex then it is also a continuous function. Observe that a decreasing function g can be k–convex only for $k \geq 1$. Moreover, when a decreasing function g is k–convex, it is also m–convex for all $m \geq k$. In the case of a strictly increasing function $g*$, it can be k–convex only for $k \leq 1$. Moreover, when $g*$ is k–convex, it is m–convex for all $m \leq k$.

Considering $k \in [1, \infty[$ we provide the following characterization given in [187].

Proposition 3.83. *Let $T : [0,1]^2 \to [0,1]$ be an Archimedean t–norm and let $g : [0,1] \to [0,\infty]$, $g(1) = 0$ be an additive generator of T. Then T is k–Lipschitz if and only if g is k–convex.*

Another useful characterization is the following

Corollary 3.84. *(Y.-H. Shyu) [224] Let $g : [0,1] \to [0,\infty]$ be an additive generator of a t–norm T which is differentiable on $]0,1[$ and let $g'(x) < 0$ for $0 < x < 1$. Then T is k–Lipschitz if and only if $g'(y) \geq kg'(x)$ whenever $0 < x < y < 1$.*

Corollary 3.85. *Let $T : [0,1]^2 \to [0,1]$ be an Archimedean t–norm and let $g : [0,1] \to [0,\infty]$ be an additive generator of T such that g is differentiable on $]0,1[\setminus S$, where $S \subset [0,1]$ is a discrete set. Then T is k–Lipschitz if and only if $kg'(x) \leq g'(y)$ for all $x, y \in [0,1], x \leq y$ such that $g'(x)$ and $g'(y)$ exist.*

Example 3.86. Consider Sugeno-Weber t–norms with an additive generator given by (p. 162)

$$g_\lambda^{SW}(t) = 1 - \frac{\log(1+\lambda t)}{\log(1+\lambda)}$$

for $\lambda \in]-1,0[$ and $\lambda \in]0,\infty[$. The derivative is

$$\frac{d}{dt} g_\lambda^{SW}(t) = -\frac{\lambda}{\log(1+\lambda)} \frac{1}{1+\lambda t}.$$

g_λ^{SW} is convex for $\lambda \geq 0$, so for these values it is a copula. For $\lambda \in]-1,0[$ the derivative reaches its minimum and maximum at $t = 1$ and $t = 0$ respectively. Thus the condition of Corollary 3.84 holds whenever $g'(1) \geq kg'(0)$. By eliminating the constant factor, we obtain

$$\frac{1+\lambda 0}{1+\lambda 1} \geq k, \text{ or } \frac{1}{1+\lambda} \geq k.$$

Therefore Sugeno-Weber t–norms are k–Lipschitz with $k = \frac{1}{1+\lambda}$ in the mentioned range. For example, $T_{-\frac{1}{2}}^{SW}$ is 2–Lipschitz. When $\lambda = -1$, this is a limiting case of the drastic (discontinuous) t–norm.

The following auxiliary results will help to determine whether a given piecewise differentiable t–norm is k–Lipschitz.

Corollary 3.87. *Let $T : [0,1]^2 \to [0,1]$ be an Archimedean t–norm and let $g : [0,1] \to [0,\infty]$ be its additive generator differentiable on $]0,1[$, and $g'(t) < 0$ on $]0,1[$. If*

$$\inf_{t \in]x,1[} g'(t) \geq k \sup_{t \in]0,x[} g'(t)$$

holds for every $x \in]0,1[$ then T is k–Lipschitz.

Proof. Follows from Corollary 3.84.

Corollary 3.88. *Let* $g : [0,1] \to [0,\infty]$ *be a strictly decreasing function, differentiable on* $]0,a[\cup]a,1[$. *If* g *is* k*-convex on* $[0,a[$ *and on* $]a,1]$, *and if*

$$\inf_{t\in]a,1[} g'(t) \geq k \sup_{t\in]0,a[} g'(t),$$

then g *is* k*–convex on* $[0,1]$.

Proof. Follows from Corollary 3.85.

Example 3.89. Consider Hamacher t–norms with an additive generator given by (p. 152)

$$g^H_\lambda(t) = \log\frac{(1-\lambda)t+\lambda}{t},$$

for $\lambda \in]0,\infty[$. The derivative is given by

$$(g^H_\lambda)'(t) = \frac{\lambda}{t(t(\lambda-1)-\lambda)}.$$

The generator g^H_λ is convex (and hence k–convex) on $[0,a]$ with $a = \frac{\lambda}{2(\lambda-1)}$. On $[a,1]$ it is k–convex, by Corollary 3.87, since the infimum of $(g^H_\lambda)'$ is reached at $t=1$ and $(g^H_\lambda)'(1) = -\lambda$, and the supremum is reached at $t=a$, and is $-4\frac{\lambda-1}{\lambda}$. Thus $k \geq \frac{\lambda^2}{4(\lambda-1)}$. Now, by Corollary 3.88, g is k–convex on $[0,1]$, with the same k, since $\sup_{t\in]0,a[}(g^H_\lambda)'(t) = (g^H_\lambda)'(a)$.

For $\lambda \in [0,2]$ T^H_λ is a copula, and for $\lambda \in [2,\infty[$ it is k–Lipschitz with $k = \frac{\lambda^2}{4(\lambda-1)}$. For instance T^H_4 is $\frac{4}{3}$-Lipschitz.

3.4.13 Calculation

Practical calculation of the values of t–norms and t–conorms is done by either a) using the two-variate expression recursively (see Fig. 1.2), or b) using the additive generators when they exist (see Fig. 3.17). In both cases one has to be careful with the limiting cases, which frequently involve infinite expressions, which result in *underflow* or *overflow* on a computer. What this means is that the generic formula should be used only for those values of the parameter λ for which numerical computation is stable.

For example, powers or exponents of λ, logarithms base λ can only be computed numerically for a restricted range of values, and this range depends on x and y as well. Consider expression t^λ in the additive generator g^{SS}_λ. We need to take into account the following limiting values $t=0, \lambda \approx 0$; $t=0, \lambda \to -\infty$; $t=0, \lambda \to \infty$; $t \approx 1, \lambda \to \pm\infty$. On a computer, the power t^λ is typically computed as $\exp(\lambda \log t)$. There is no point to compute $\exp(t)$ for $t < -20$ or $t > 50$, as the result is smaller than the machine epsilon or is too large.

When performing calculations using additive generators, one has to exercise care with the values of t near 0 (and near 1 for t–conorms), as strict t–norms have an asymptote at this point. The value of 0 (or 1 for t–conorms) should be returned as a special case and evaluation of the additive generator skipped.

For the purposes of numerical stability, it is possible to scale the additive generator by using a positive factor (we remind that it is defined up to an arbitrary positive factor). This does not affect the numerical value of the t–norm or t–conorm.

```
typedef double ( *USER_FUNCTION)( double );
double p=1.0, eps=0.001;

/* example of an additive generator with parameter p (Hamacher) */
double g(double t)
{     return log( ((1-p)*t + p)/t );  }
double ginv(double t)
{    return p / (exp(t)+p-1); }

double f_eval(int n, double * x, USER_FUNCTION g, USER_FUNCTION gi)
{
   int i;
   double r=0;
   for(i=0;i<n;i++) {
      if(x[i]<= eps) return 0.0;
      r+= g(x[i]);
   }
   return gi(r);
}
...

/* calling the function */
x[0]=0.1; x[1]=0.2;
double z= f_eval(2,x,&g,&ginv);
```

Fig. 3.17. A C++ code for evaluation of an Archimedean t–norm using its additive generator.

3.4.14 How to choose a triangular norm/conorm

In this chapter we have seen that there are many different conjunctive/ disjunctive aggregation functions, in particular t–norms and t–conorms. In fact, there are several infinite families of such functions, and it is easy to construct

even more, by either modifying the additive generators, or using the ordinal sum construction. The question we want to answer in this section is how to choose the most suitable aggregation function for a specific application.

The first thing one has to do is to determine all the required application-specific properties, which would narrow down the choice. For example, are associativity and symmetry required? Are there nilpotent elements? How strong is mutual reinforcement of the arguments?

Still even after limiting the choices, there are infinitely many functions that satisfy application requirements. Then it comes down to using some numerical data. We shall use a general approach discussed in Section 1.6. Let us have a set of empirical data, pairs $(\mathbf{x}_k, y_k), k = 1, \ldots, K$, which we want to fit as best as possible by using an aggregation function from a given class, such as t–norm or t–conorm. Our goal is to determine the best function from that class that minimizes the norm of the differences between the predicted $(f(\mathbf{x}_k))$ and observed (y_k) values. We will use the least squares or least absolute deviation criterion, as discussed on p. 33.

Depending on the class of aggregation functions, we have two choices: a) if the class is a parametric family of functions (e.g., a given family of t-norms), then we need to determine the best value of such a parameter; b) if the class is more general (e.g, all continuous t–norms) then we need to consider non-parametric methods, which we will discuss in Section 3.4.15.

Let us concentrate on fitting a parameter of a given family of t–norms. The case of t–conorms is reduced to that of t–norms by using duality: consider an auxiliary data set $\tilde{\mathcal{D}} = \{\tilde{\mathbf{x}}_k, \tilde{y}_k\}_{k=1}^K$, where $\tilde{x}_{ik} = 1 - x_{ik}$, $\tilde{y}_k = 1 - y_k$. Fit a t–norm T to that data set. The desired t–conorm is the dual of T.

Take a family of t–norms, say the Yager family (p. 156). Fitting of a nonlinear parameter λ in the least squares sense involves solving the following optimization problem

$$\min_{\lambda} \sum_{k=1}^{K} \left(T_\lambda^Y(\mathbf{x}_k) - y_k\right)^2, \tag{3.14}$$

λ unrestricted. In the case of the least absolute deviation criterion (LAD) we minimize

$$\min_{\lambda} \sum_{k=1}^{K} \left|T_\lambda^Y(\mathbf{x}_k) - y_k\right|. \tag{3.15}$$

This is a typical nonlinear optimization problem (smooth (3.14) or non-smooth (3.15)). There is no guarantee that the objective function is convex, or has a unique global minimum. Therefore we recur to methods of global optimization, discussed in the Appendix A.5.

When using numerical solutions, one has to bear in mind the specifics of the problem. 1) Large and small values of λ lead to the special cases, refer to a specific family of t–norms. 2) Evaluation of t–norms requires bounding the range of λ to avoid numerical instability. Consequently, one effectively solves a univariate global optimization problem on one or more bounded intervals,

and replaces the generic formula with the limiting cases on the rest of the domain.

The methods of choice are grid search with subsequent local descent (by using a derivative free non-smooth optimization method), or Pijavski-Shubert deterministic method, also with subsequent improvement by local descent.

The associativity property of t–norms allows one to formulate a more general data fitting problem. Remember that a family of t–norms is an extended aggregation function, it is defined for any number of inputs. It is quite feasible that the same t–norm will be used in an application to aggregate different numbers of inputs. Is it possible to use empirical data of varying dimension to fit the whole extended aggregation function (i.e., all n-variate aggregation functions, $n = 2, 3, \ldots$?

Thus we consider the data set $\mathcal{D}$ in which input vectors $\mathbf{x}_k$ may have different dimension, denoted by n_k, as illustrated in Table 3.2.

Table 3.2. A data set with inputs of varying dimension.

k	n_k	x_1	x_2	x_3	x_4	x_5	y
1	3	x_{11}	x_{21}	x_{31}			y_1
2	2	x_{12}	x_{22}				y_2
3	3	x_{13}	x_{23}	x_{33}			y_3
4	5	x_{14}	x_{24}	x_{34}	x_{44}	x_{54}	y_4
5	4	x_{15}	x_{25}	x_{35}	x_{45}		y_5
$\vdots$							

Interestingly, the optimization problems (3.14) or (3.15) need no modification, as long as the t–norm T_λ^Y is calculated consistently for any number of arguments. The resulting optimal parameter λ defines an optimal aggregation function from the chosen family.

3.4.15 How to fit additive generators

It is often impossible to decide, based on application-specific requirements, which particular parametric family of t–norms is the most suitable. We will investigate the problem of fitting *an arbitrary* continuous t–norm to the data, and consider separately strict and nilpotent t–norms. We know from Section 3.4.10 that any continuous t–norm can be approximated uniformly and with any desired accuracy by a continuous Archimedean t–norm. In turn, to fit a continuous Archimedean t–norm, we need to fit its additive generator, which is a strictly monotone univariate function. The approach we explore in this section is how to fit additive generators, which is effectively a tool for fitting arbitrary continuous t–norms.

There are no specific requirements on an additive generator g, besides monotonicity and satisfying $g(1) = 0$, $g(0.5) = 1$. We remind that the latter

condition is used to fix a specific additive generator, because it is not defined uniquely. The number 0.5 can be replaced with any other number in $]0,1[$. An additive generator needs not be differentiable, or have any specific algebraic form.

The method of spline approximation is very popular in numerical approximation, as splines are very flexible to model functions of any shape (see Appendix A.1). An additional advantage is that polynomial splines allow one to represent the condition of monotonicity via a simple set of linear inequalities involving spline coefficients [13, 15]. We will use regression splines, defined as a linear combination of B-splines with a priori fixed knots $t_1, t_2, \ldots, t_m$ in the interval $[0,1]$,

$$S(t) = \sum_{j=1}^{J} c_j B_j(t), \tag{3.16}$$

with coefficients c_j to be determined from the data. B_j are usually chosen as B-splines, although other choices are possible [77].

The simplest spline (or degree 1) is the broken line approximation, where the data points are joined by straight lines (Figure A.1). The precision of spline approximation is easily controlled by increasing the number of knots, and because of their extremal properties, polynomial splines are considered the "smoothest" curves that interpolate or approximate the data. Monotone regression splines are explored in detail in [13, 15], where the condition of monotonicity is reduced to that of non-negativity (non-positivity) of coefficients c_j in a suitably chosen basis. The spline regression problem is formulated as a linearly constrained least squares problem, which can be solved by a variety of methods [152], see Appendix A.3.

Our approach to construction of continuous t–norms consists in using a monotone regression spline S in (3.16) as an additive generator. We shall use the data set $\mathcal{D}$ in which input vectors $\mathbf{x}_k$ may have different dimensions, as in Table 3.2. By applying Eq. (3.7) and the least squares criterion, we solve

$$\text{Minimize} \sum_{k=1}^{K} \left(S(x_{1k}) + S(x_{2k}) + \ldots + S(x_{n_k k}) - S(y_k)\right)^2, \tag{3.17}$$

subject to the conditions that S is monotone decreasing, $S(1) = 0$ and $S(a) = 1$. Convenient choices of the value $a \in]0,1[$ will be discussed later in this section.

Conditions of monotonicity translate into $c_j < 0$ for the B-spline basis in [13]. Equality conditions become linear equality constraints

$$S(1) = \sum_{j=1}^{J} c_j B_j(1) = 0, \quad S(a) = \sum_{j=1}^{J} c_j B_j(a) = 1.$$

Replacing S with (3.16) and rearranging the terms in the sum, we obtain a quadratic programming problem

$$\begin{aligned}
\text{Minimize} \quad & \sum_{k=1}^{K}\left(\sum_{j=1}^{J} c_j\left[B_j(x_{1k})+B_j(x_{2k})+\ldots+B_j(x_{n_k k})-B_j(y_k)\right]\right)^2 \\
\text{s.t.} \quad & \sum_{j=1}^{J} c_j B_j(1)=0, \\
& \sum_{j=1}^{J} c_j B_j(a)=1, \\
& c_j<0.
\end{aligned} \tag{3.18}$$

For numerical purposes, the strict inequalities are converted to $c_j \le -\varepsilon < 0$ for some small ε. Observe that the expression in the square brackets can be written as a single function $B_j(\mathbf{x}, y)$ to simplify the notation. We also note that problem (3.18) is of type LSEI (Appendix A.3, Eq. (A.8)), for which special methods have been designed.

In the case of the least absolute deviation criterion, we obtain an optimization problem, subsequently converted to a linear programming problem (Eq. (A.4) in the Appendix), namely

$$\begin{aligned}
\text{Minimize} \quad & \sum_{k=1}^{K}\left|\sum_{j=1}^{J} c_j B_j(\mathbf{x}_k, y_k)\right| \\
\text{s.t.} \quad & \sum_{j=1}^{J} c_j B_j(1)=0, \\
& \sum_{j=1}^{J} c_j B_j(a)=1, \\
& c_j<0.
\end{aligned} \tag{3.19}$$

Our next task is to determine a. We distinguish two cases: nilpotent and strict t–norms. For nilpotent t–norms, whose additive generators satisfy $g(0) < \infty$, the choice is simple, any value of a will do, so we use $a = 0.5$ for simplicity. However, for strict t–norms we need to model asymptotic behavior near 0. Polynomial splines are not suitable, as they are finite. Furthermore, the usual trick of replacing ∞ with a large finite number does not work, because additive generators are defined up to a positive multiplier. That is, setting $S(0) = 1000$ is equivalent to $S(0) = 1$ or any other number, as this number is factored out from the objective function in (3.18) or (3.19).

A workaround is to use well-founded additive generators [136, 137], defined as

$$g(t) = \begin{cases} \frac{1}{t} + S(\varepsilon) - \frac{1}{\varepsilon}, & \text{if } t \le \varepsilon, \\ S(t), & \text{if } t > \varepsilon. \end{cases} \tag{3.20}$$

In this case we set $a = \varepsilon$, where ε is the smallest strictly positive value among $x_{ik}, y_k, i = 1, \ldots, n_k, k = 1 \ldots, K$.

The asymptote near 0 is modeled by the function $1/t$. The reason why we can use this function (or, in fact, any other function with the same asymptotic

behavior) is that no observed data falls within $]0, \varepsilon[$. Consequently the values of the additive generator g on $]0, \varepsilon[$ are not used to calculate any quantity in problems (3.18) or (3.19), and can be chosen with relative freedom, as long as continuity and monotonicity of g are kept (see [17], p. 915).

Finally, differentiability of an additive generator can be imposed by using quadratic monotone regression splines. Quadratic B-splines are easily calculated using recursive equations (see Appendix A.1), and conditions of monotonicity are quite similar to those for linear splines [13].

Figure 3.18 illustrates approximation of an additive generator by linear splines. The data $(\mathbf{x}_k, y_k)$ were generated by using Dombi t–norm $T^D_{0.2}$, whose additive generator is given by

$$g(t) = \left(\frac{1-t}{t}\right)^{0.2}.$$

These data were used to solve problem (3.18), with the well-founded additive generator. Fig. 3.18 shows the original additive generator and its linear spline approximation (left), and the actual approximation to the t–norm (right).

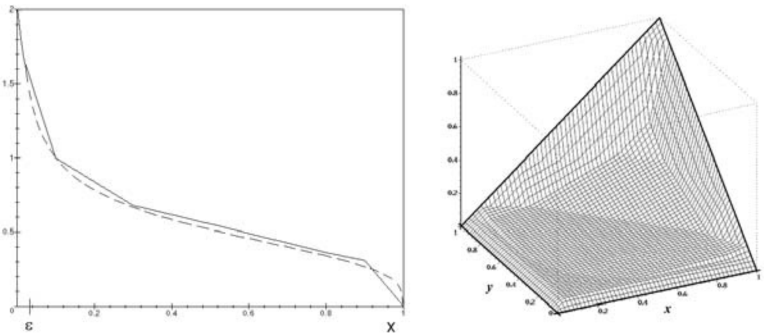


Fig. 3.18. Approximation of the Dombi t-norm $T^D_{0.2}$ (right) and of its additive generator (left), using 20 randomly generated data. The spline approximation is the solid piecewise linear curve.

Preservation of ordering of the outputs

We recall from Section 1.6, p. 34, that sometimes one has to fit not only an aggregation function to numerical data, but also preserve the ordering of the outputs. That is, if $y_j \le y_k$ then we expect $f(\mathbf{x}_j) \le f(\mathbf{x}_k)$. We already discussed approaches to preservation of output ordering when dealing with averaging aggregation functions. For t–norms, we can rely on a very similar technique.

First, arrange the data, so that the outputs are in non-decreasing order, i.e., $y_k \leq y_{k+1}, k = 1, \ldots, K-1$. Define the additional linear constraints

$$\sum_{j=1}^{J} c_j \left[B(x_{1k}) + \ldots + B(x_{n_k k}))\right] - \left[B(x_{1k+1}) + \ldots + B(x_{n_{k+1}k+1}))\right] \geq 0,$$

$k = 1, \ldots, K-1$. The change of the sign of the inequality is due to the fact that S is decreasing. Add the above constraints to problem (3.18) or (3.19) and solve it.

The addition of the extra $K-1$ constraints does not change the structure of the optimization problem, nor drastically affect its complexity.

Approximation of copulas

One important subclass of Archimedean t-norms is Archimedean copulas, see Section 3.5. Archimedean copulas are characterized by convex additive generators. Convexity of splines in (3.16) is imposed by adding extra linear constraints on the coefficients $c_{j+1} - c_j \geq 0, j = 1, \ldots, J-1$.

3.4.16 Introduction of weights

There were numerous studies of the issue of introducing weights into conjunctive and disjunctive aggregation functions, and in particular t-norms and t-conorms [47, 48, 49, 84, 261, 272] (and references therein). Weighting vectors played a very important role in Chapter 2 in the context of various means. Weights represented such concepts as the importance of the criteria, importance of an expert's opinion, or quality and reliability of information sources. It turns out that similar techniques are applicable to t-norms and t-conorms, as well as various mixed type aggregation functions discussed in Chaper 4.

We briefly outline one such process, applicable to Archimedean t-norms and t-conorms, which allows one to obtain weighted versions of these aggregation functions.

First, let us establish the fundamental properties of weighted aggregation. Let T be a t-norm and $T_{\mathbf{w}}$ be its weighted counterpart. The vector of weights $\mathbf{w}$ must have non-negative components, but we do not require its normalization, like $\sum w_i = 1$, we simply have $w_i \geq 0$.

- If all weights $w_i = 1$, then $T_{\mathbf{w}} = T$.
- If any $w_i = 0$, then the i-th input is irrelevant and $T_{\mathbf{w}}(\mathbf{x}) = T_{\hat{\mathbf{w}}}(\hat{\mathbf{x}})$, where $\hat{\mathbf{w}}, \hat{\mathbf{x}}$ are obtained from $\mathbf{w}, \mathbf{x}$ by removing the i-th component.

Note 3.90. Since t-norms have neutral element $e = 1$, we have an interesting counterpart of the second property, $T_{\mathbf{w}}(\mathbf{x}) = T_{\mathbf{w}}(\tilde{\mathbf{x}})$, where $\tilde{\mathbf{x}}$ is a vector obtained from $\mathbf{x}$ by replacing those components that correspond to zero weights with ones. For example, $T_{(0,w_2,w_3)}(x_1, x_2, x_3) = T_{(w_1,w_2,w_3)}(1, x_2, x_3) = T_{(w_2,w_3)}(x_2, x_3)$.

Consider now continuous Archimedean t-norms, which are expressed via additive generators g as

$$T(\mathbf{x}) = g^{(-1)}\left(\sum_{i=1}^{n} g(x_i)\right).$$

The method proposed in [83, 261] consists in replacing this expression with its weighted analogue, defined as follows.

Definition 3.91 (Weighted Archimedean t-norms). *Let $g : [0,1] \to [0,\infty]$, $g(1) = 0$ be an additive generator of some Archimedean t-norm T, and $\mathbf{w} : w_i \geq 0$ be a (not necessarily normalized) weighting vector. The weighted Archimedean t-norm is defined as*

$$T_{\mathbf{w}}(\mathbf{x}) = g^{(-1)}\left(\sum_{i=1}^{n} w_i g(x_i)\right). \tag{3.21}$$

Note 3.92. In [83] there was an additional constraint $\sum\limits_{i=1}^{n} w_i = n$, but it was not included in the later studies.

Note 3.93. Weighted t-norms may acquire averaging behavior, in particular they convert into weighted quasi-arithmetic means if $\sum\limits_{i=1}^{n} w_i = 1$. However, weighted t–norms are **not conjunctive** aggregation functions, unless $w_i \geq 1, i = 1, \ldots, n$.

Example 3.94. [261] Weighted product t-norm ($g = \log$) is given as

$$T_{P,\mathbf{w}} = \prod_{i=1}^{n} x_i^{w_i}.$$

Note the similarity to the geometric mean $G_{\mathbf{w}}$, in which case the weighting vector needs to be normalized by $\sum\limits_{i=1}^{n} w_i = 1$.

Example 3.95. Weighted Łukasiewicz t-norm is given by

$$T_{L,\mathbf{w}}(\mathbf{x}) = \max(0, 1 - \sum_{i=1}^{n} w_i(1 - x_i)).$$

Example 3.96. Weighted Yager t-norm is given by

$$T^{Y}_{\lambda,\mathbf{w}}(\mathbf{x}) = \max(0, (1 - \sum_{i=1}^{n} w_i(1 - x_i)^{\lambda})^{1/\lambda}).$$

Example 3.97. Weighted Frank t-norm is given by

$$T^F_{\lambda,\mathbf{w}}(\mathbf{x}) = \log_\lambda \left(1 + \frac{\prod_{i=1}^{n} (\lambda^{x_i} - 1)^{w_i}}{(\lambda - 1)^{\sum_{i=1}^{n} w_i - 1}} \right).$$

By using duality, we obtain a similar construction for weighted t-conorms.

Definition 3.98 (Weighted Archimedean t-conorms). *Let $h : [0,1] \to [0,\infty]$, $h(0) = 0$ be an additive generator of some Archimedean t-conorm S, and $\mathbf{w} : w_i \geq 0$ be a (not necessarily normalized) weighting vector. The weighted Archimedean t-conorm is defined as*

$$S_{\mathbf{w}}(\mathbf{x}) = h^{(-1)} \left(\sum_{i=1}^{n} w_i h(x_i) \right). \tag{3.22}$$

Example 3.99. Referring to Examples 3.94-3.97 we have

$$S_{P,\mathbf{w}} = 1 - \prod_{i=1}^{n} (1 - x_i)^{w_i},$$

$$S_{L,\mathbf{w}}(\mathbf{x}) = \min(1, \sum_{i=1}^{n} w_i x_i),$$

$$S^Y_{\lambda,\mathbf{w}}(\mathbf{x}) = \min(1, (\sum_{i=1}^{n} w_i (1 - x_i)^\lambda)^{1/\lambda}),$$

$$S^F_{\lambda,\mathbf{w}}(\mathbf{x}) = 1 - \log_\lambda \left(1 + \frac{\prod_{i=1}^{n} (\lambda^{1-x_i} - 1)^{w_i}}{(\lambda - 1)^{\sum_{i=1}^{n} w_i - 1}} \right).$$

Yager [272] provided an interesting view of the above mentioned process. Let us introduce a bivariate function $H : [0,1]^2 \to [0,1]$, called the *importance transformation function*, defined by

$$H(w,t) = g^{(-1)}(wg(t)).$$

Then we can express (3.21) as

$$T_{\mathbf{w}} = T(H(w_1,x_1), \ldots, H(w_n,x_n)),$$

provided that $w_i \in [0,1]$, but not necessarily sum to one [13].

The function H satisfies the following properties:

[13] In fact, restrictions $w_i \leq 1$ are not necessary, H can be defined on $[0,\infty] \times [0,1]$. However, it looses its interpretation as an implication function, see footnote 14.

- $H(1,t) = g^{(-1)}(g(t)) = t$;
- $H(0,t) = g^{(-1)}(0) = 1$;
- $H(w,t)$ is non-decreasing in t and is non-increasing in w.

Note 3.100. The above mentioned properties are exactly the properties of some *implication* functions [14], notably S- and R-implications. Then we can have an alternative definition of weighted t-norms, starting from an arbitrary S- or R- implication I, as

$$T_{\mathbf{w}} = T(I(w_1, x_1), \ldots, I(w_n, x_n)).$$

Note 3.101. Yager [272] suggests that the importance transformation function H may be defined using a different additive generator $\hat{g}$ from that of the t-norm T, i.e., $H(w,t) = \hat{g}^{(-1)}(w\hat{g}(t))$.

For weighted t-conorms we have an analogous expression

$$S_{\mathbf{w}} = S(\hat{H}(w_1, x_1), \ldots, \hat{H}(w_n, x_n)),$$

with function $\hat{H}(w,t) = h^{(-1)}(wh(t))$. However, now h is an additive generator of a t-conorm, and is increasing, hence function $\hat{H}$ has a different set of properties compared to H, making it inconsistent with implication functions. Namely $\hat{H}$ is non-decreasing in both arguments and satisfies $\hat{H}(0,t) = 0$. By using a strong negation N one can represent it via an implication function as $\hat{H}(w,t) = N(I(w, N(t))$. For S-implication it yields $\hat{H}(w,t) = N(S(N(w), N(t)) = T(w,t)$, the t-norm dual to S.

Example 3.102. Let S be max, S_1 be some t-conorm which generates an S-implication I, and $\mathbf{w}$ be a weighting vector. Using $\hat{H}(w,t) = N(I(w, N(t))$ we obtain

$$S_{\mathbf{w}} = \max\{\hat{H}(w_1, x_1), \ldots, \hat{H}(w_n, x_n)\} = \max\{T_1(w_1, x_1), \ldots, T_1(w_n, x_n)\},$$

where T_1 is the t-norm dual to S_1. This is a well-known weighted max function [83, 97, 269].

[14] A bivariate function $I : [0,1]^2 \to [0,1]$ is called an implication function [98, 172] if:

- I is non-decreasing in the first variable;
- I is non-increasing in the second variable;
- $I(0,0) = I(0,1) = I(1,1) = 1$ and $I(1,0) = 0$.

S-implications are defined by $I_S(x,y) = S(N(x), y)$, where S is a t-conorm and N is a strong negation. An R-implication is defined by $I_T(x,y) = \sup\{z \in [0,1] | \ T(x,z) \le y\}$ where T is a left-continuous t-norm. There are also other implications, see [98, 172].

Example 3.103. Let $T = \min$, S be some t-conorm, and let $\mathbf{w}$ be a weighting vector. Using the same argument we obtain a weighted t-conorm

$$S_{\mathbf{w}} = S(\hat{H}(w_1, x_1), \ldots, \hat{H}(w_n, x_n)) = S(\min(w_1, x_1), \ldots, \min(w_n, x_n)).$$

Of course, we can also use any other t-norm instead of min. A further generalization is obtained by using [47]

$$S_{\mathbf{w}} = S(A(w_1, x_1), \ldots, A(w_n, x_n)),$$

where $A : [0, \infty[\times[0, 1] \to [0, 1]$ is the transformation function defined by

$$A(w,t) = \sup\left\{ y \in [0,1] |\ \exists i, j \in \{1,2,\ldots\}, \frac{i}{j} < w \text{ and } w \in [0,1] \right.$$
$$\left. \text{such that } S(\overbrace{u,\ldots,u}^{j-times}) < t \text{ and } y = S(\overbrace{u,\ldots,u}^{i-times}) \right\}.$$

A is a generalization of the usual multiplication, see [47] for details.

Other weighted aggregation functions based on t-norms and t-conorms are considered in [47, 272].

Construction based on composition

Let us now consider an alternative construction method, based on a composition of an averaging aggregation function f, such as a weighted quasi-arithmetic mean or Choquet integral discussed in Chapter 2, and a non-decreasing function $\psi : [0,1] \to [0,1], \psi(0) = 0, \psi(1) = 1$, using Proposition 1.85. Under suitable conditions, namely $\psi(f(1,\ldots,1,t,1,\ldots,1)) \le t$ for all $t \in [0,1]$ and at any position, the composition $\psi \circ f$ is a *conjunctive* aggregation function (which was not generally the case for weighted t–norms). Consider the following examples.

Example 3.104. Let f be a quasi-arithmetic mean with a strictly decreasing generating function g, such that $g(1) = 0$ (we remind that generating functions are defined up to a linear transformation, see Section 2.3.2, so such choice is possible if 1 is not an absorbing element of f). Then condition $\psi(f(1,\ldots,1,t,1,\ldots,1)) \le t$ entails

$$\psi(g^{-1}(g(t)/n)) \le t.$$

Take $\psi(t) = g^{(-1)}(ng(t)) = g^{-1}(\min(g(0), ng(t)))$. Then we have $\psi(0) = 0, \psi(1) = 1$ and

$$\psi(g^{-1}(g(t)/n)) = g^{-1}(\min(g(0), g(t)) = t.$$

Consequently, $\psi \circ f$ is a conjunctive aggregation function with the neutral element $e = 1$.

Example 3.105. A specific instance of Example 3.104 is the case of power means

$$g(t) = \begin{cases} t^r - 1, & \text{if } r < 0, \\ -(t^r - 1), & \text{if } r > 0, \\ -\log(t), & \text{if } r = 0. \end{cases}$$

For $r \neq 0$ define $\psi(t) = (\max(0, nt^r - (n-1)))^{1/r}$ and $\psi(t) = t^n$ for $t = 0$. The function $\psi \circ f$ becomes

$$(\psi \circ f)(\mathbf{x}) = \begin{cases} \max(0, \sum\limits_{i=1}^{n} x_i^r - (n-1))^{1/r}, & \text{if } r \neq 0, \\ \prod\limits_{i=1}^{n} x_i, & \text{if } r = 0, \end{cases}$$

which is the Schweizer-Sklar family of triangular norms, p. 150. On the other hand, if we take $g(t) = (1-t)^r, r > 0$, we can use $\psi = \max(0, 1 - n^{1/r}(1-t))$, and in this case we obtain

$$(\psi \circ f)(\mathbf{x}) = \max(0, 1 - (\sum_{i=1}^{n} (1 - x_i)^r)^{1/r}),$$

i.e., the Yager family of triangular norms, p. 156.

Example 3.106. Consider a weighted quasi-arithmetic mean with a strictly decreasing generating function g, such that $g(1) = 0$, and strictly positive weighting vector $\mathbf{w}$. As in Example 3.104, take $\psi(t) = g^{(-1)}(\frac{1}{\min w_i} g(t)) = g^{-1}(\min(g(0), \frac{1}{\min w_i} g(t)))$. Then we have $\psi(0) = 0, \psi(1) = 1$ and for all $j = 1, \ldots, n$

$$\psi(g^{-1}(w_j g(t))) = g^{-1}(\min(g(0), \frac{w_j}{\min w_i} g(t)) \leq t.$$

Consequently, $\psi \circ f$ is a conjunctive aggregation function for a given f. Moreover, it is the strongest conjunctive function of the form $\psi \circ f$. It depends on the weighting vector $\mathbf{w}$, but it differs from the weighted t–norms. Note that such a weighted conjunctive function can be written with the help of the importance transformation function $H : [0, \infty] \times [0, 1] \to [0, 1]$ (see p. 178) as

$$T_{\tilde{\mathbf{w}}}(\mathbf{x}) = T(H(\tilde{w}_1, x_1), \ldots, H(\tilde{w}_n, x_n))$$

with the modified vector $\mathbf{w} = (\frac{w_1}{\min w_i}, \ldots, \frac{w_n}{\min w_i})$.

Example 3.107. As a specific instance, consider weighted arithmetic mean $M_{\mathbf{w}}$ with a strictly positive weighting vector $\mathbf{w}$. Take $\psi(t) = \max(0, \frac{t}{w^*} - (\frac{1}{w^*} - 1))$, with $w^* = \min w_i$. Then the corresponding weighted conjunctive function is

$$(\psi \circ f)(\mathbf{x}) = \max(0, \frac{1}{w^*}(M_{\mathbf{w}}(\mathbf{x}) - 1 + w^*))$$

In the same way weighted conjunctive aggregation functions can be constructed from other averaging functions such as OWA and Choquet integrals treated in Sections 2.5 and 2.6.

3.5 Copulas and dual copulas

The problem of the construction of distributions with given marginals can be reduced, thanks to Sklar's theorem [228], to construction of a copula. There exist different methods for constructing copulas, presented, for instance, in [195, 196]. We proceed with the definition of bivariate copulas.

Definition 3.108 (Copula). *A bivariate copula is a function $C : [0,1]^2 \to [0,1]$ which satisfies:*

- $C(x,0) = C(0,x) = 0$ *and* $C(x,1) = C(1,x) = x$ *for all* $x \in [0,1]$ *(boundary conditions);*
- $C(x_1,y_1) - C(x_1,y_2) - C(x_2,y_1) + C(x_2,y_2) \geq 0$, *for all* $x_1,y_1,x_2,y_2 \in [0,1]$ *such that* $x_1 \leq x_2$ *and* $y_1 \leq y_2$ *(2–increasing property).*

In statistical analysis, a joint distribution H of a pair of random variables (X,Y) with marginals F and G respectively, can be expressed by $H(x,y) = C(F(x),G(y))$ for each $(x,y) \in [-\infty,\infty]^2$, where C is a copula uniquely determined on $Ran(F) \times Ran(G)$.

Example 3.109.

- The product, Łukasiewicz and minimum t–norms are copulas.
- The function $C_u : [0,1]^2 \to [0,1]$ defined for each $u \in [0,1)$ as follows

$$C_u(x,y) = \begin{cases} \max(x+y-1,u), & (x,y) \in [u,1]^2, \\ \min(x,y) & \text{otherwise} \end{cases}$$

  is also a copula.

Main properties

- Copulas are monotone non-decreasing (and thus are aggregation functions);
- Copulas verify $T_L \leq C \leq \min$ (and thus are conjunctive functions);
- Copulas are not necessarily symmetric or associative;
- Copulas satisfy the 1-Lipschitz property, and are hence continuous;
- A convex combination of copulas is a copula.
- A t-norm is a copula if and only if it is 1-Lipschitz.

Note 3.110. **Every associative** copula is a continuous t–norm [142], p.204. But not every t-norm is an associative and symmetric copula (take T_D as a counterexample). Not every copula is a t–norm, for instance $C(x,y) = xy + x^2y(1-x)(1-y)$ is an asymmetric copula and therefore not a t–norm.

Definition 3.111. *Let C be a copula.*

- *The function* $\tilde{C}(x,y) : [0,1]^2 \to [0,1]$ *given by*

$$\tilde{C}(x,y) = x + y - C(x,y)$$

is called the dual of the copula C (it is not a copula);

- *The function* $C^*(x,y) : [0,1]^2 \to [0,1]$ *given by*

$$C^*(x,y) = 1 - C(1-x, 1-y)$$

is called the co–copula of C (it is not a copula);
- *The function $C^t(x,y) = C(y,x)$ is also a copula, called the transpose of C. A copula is symmetric if and only if $C^t = C$.*

Note 3.112. For historical reasons the term "dual copula" does not refer to the dual in the sense of Definition 1.54 of the dual aggregation function. Co–copula is the dual in the sense of Definition 1.54.

Note 3.113. The dual of an associative copula, $\tilde{C}$, is not necessarily associative. The dual copulas of Frank t-norms (p. 154) (which are copulas themselves), are also associative (but disjunctive) symmetric aggregation functions.

If C is a bivariate copula, then the following functions are also copulas:

- $C_1(x,y) = x - C(x, 1-y)$;
- $C_2(x,y) = y - C(1-x, y)$;
- $C_3(x,y) = x + y - 1 + C(1-x, 1-y)$;

The concept of a quasi–copula (Definition 3.8 on p. 125) is more general than that of a copula. This was introduced by Alsina et al. in 1983 [5]. Each copula is a quasi–copula but the converse is not always true. They verify $T_L \leq Q \leq \min$ for all quasi–copulas Q. Any copula can be uniquely determined by means of its diagonal (see [195]).

Another related concept is semicopula (Definition 3.6), which includes the class of quasi–copulas and hence the class of copulas.

Note 3.114. Note the following statements:

- A semicopula C that satisfies the 2–increasing condition $C(x_1,y_1) - C(x_1,y_2) - C(x_2,y_1) + C(x_2,y_2) \geq 0$, for all $x_1, y_1, x_2, y_2 \in [0,1]$ such that $x_1 \leq x_2$ and $y_1 \leq y_2$ is a copula.
- A semicopula Q that satisfies the Lipschitz condition $|Q(x_1,y_1) - Q(x_2,y_2)| \geq |x_1 - x_2| + |y_1 - y_2|$, is a quasi–copula.
- A semicopula T that is both symmetric and associative is a t–norm.

Example 3.115. The drastic product is a semicopula but not a quasi—copula. The function $S(x,y) = xy\max(x,y)$ is a semicopula but is not a t-norm because it is not associative. However, the function

$$T(x,y) = \begin{cases} 0, & \text{if } (x,y) \in [0,1/2] \times [0,1[, \\ \min(x,y) & \text{otherwise} \end{cases}$$

is not a quasi–copula (not Lipschitz), but is a semicopula, since it has neutral element $e = 1$.

Archimedean copulas

An important class of copulas are the Archimedean copulas. They are useful for several reasons: a) they can be constructed easily; b) a wide variety of copulas belong to this class; c) copulas in this class have special properties. Archimedean copulas appeared, for the first time, in the study of probabilistic metric spaces.

Archimedean copulas are characterized by convex additive generators [15].

Proposition 3.116. *Let $g : [0,1] \to [0,\infty]$ be a continuous strictly decreasing function with $g(1) = 0$, and let $g^{(-1)}$ be the pseudo–inverse of g. The function $C : [0,1]^2 \to [0,1]$ given by*

$$C(x,y) = g^{(-1)}(g(x) + g(y)) \tag{3.23}$$

is a copula if and only if g is convex.

This result gives us a way to construct copulas, for that we only need to find continuous, strictly decreasing and convex functions g from $[0,1]$ to $[0,\infty]$, with $g(1) = 0$, and then to define the copula by $C(x,y) = g^{(-1)}(g(x) + g(y))$. For instance if $g(t) = \frac{1}{t} - 1$ we have $C(x,y) = \frac{xy}{x+y-xy}$. For short we will denote this copula by $C = \frac{\Pi}{\Sigma - \Pi}$.

Definition 3.117. ***(Archimedean copula)*** *A copula given by (3.23), with $g : [0,1] \to [0,\infty]$ being a continuous strictly decreasing function with $g(1) = 0$, is called Archimedean. The function g is its additive generator. If $g(0) = \infty$, we say that g is a strict generator, then $g^{(-1)} = g^{-1}$ and $C(x,y) = g^{-1}(g(x) + g(y))$ is called a strict Archimedean copula. If $g(0) < \infty$, then C is a nilpotent copula. If the function g is an additive generator of C, then the function $\theta(t) = e^{-g(t)}$, is a multiplicative generator of C.*

Note 3.118. Archimedean copulas are a subclass of continuous Archimedean t-norms (those with a convex additive generator).

[15] A function g is convex if and only if $g(\alpha t_1 + (1-\alpha)t_2) \le \alpha g(t_1) + (1-\alpha)g(t_2)$ for all $t_1, t_2 \in Dom(g)$ and $\alpha \in [0,1]$.

Example 3.119.

1. $C(x,y) = xy$ is a strict Archimedean copula with an additive generator $g(t) = -\log t$.
2. $C(x,y) = \max(x+y-1,0)$ is also an Archimedean copula with an additive generator $g(t) = 1-t$.
3. $C(x,y) = \min(x,y)$ is not an Archimedean copula.
4. $C(x,y) = xye^{-a \log x \log y}$ is an Archimedean copula for $a \in (0,1]$, with an additive generator $g(t) = \log(1 - a \log t)$, called Gumbel-Barnett copula [196]. When $a = 0$, $C(x,y) = xy$.

Note 3.120. Among many properties of Archimedean copulas C, with additive generators g, we will quote the following[16]:

- C is symmetric;
- C is associative;
- If $c > 0$ is a real number then $c \cdot g$ is also an additive generator of the same copula.
- Archimedean property: for any $u, v \in]0,1[$, there is a positive integer n such that $C(\overbrace{u, \ldots, u}^{n-times}) < v$.

We summarize different families of parameterized copulas in Table 3.3. The limiting cases of these copulas are presented in Table 3.4. For a comprehensive overview of copulas see [196].

Table 3.3. Some parameterized families of copulas.

#	$C_\lambda(x,y)$	$g_\lambda(t)$	$\lambda \in$	strict
1.	$\max([x^{-\lambda} + y^{-\lambda} - 1]^{-\frac{1}{\lambda}}, 0)$	$\frac{1}{\lambda}(t^{-\lambda} - 1)$	$[-1,\infty) \setminus \{0\}$	$\lambda \geq 0$
2.	$\max(1 - [(1-x)^\lambda + (1-y)^\lambda]^{\frac{1}{\lambda}}, 0)$	$(1-t)^\lambda$	$[1,\infty)$	no
3.	$\frac{xy}{1-\lambda(1-x)(1-y)}$	$\log(\frac{1-\lambda(1-t)}{t})$	$[-1,1)$	yes
4.	$e^{(-((-\log x)^\lambda + (-\log y)^\lambda)^{\frac{1}{\lambda}})}$	$(-\log t)^\lambda$	$[1,\infty)$	yes
5.	$-\frac{1}{\lambda}\log(1 + \frac{(e^{-\lambda x}-1)(e^{-\lambda y}-1)}{e^{-\lambda}-1})$	$-\log(\frac{e^{-\lambda t}-1}{e^{-\lambda}-1})$	$(-\infty,\infty) \setminus \{0\}$	yes

The family of copulas #1 is called Clayton family, the family #3 is called Ali-Mikhail-Haq family (they belong to Hamacher family of t–norms), the family #4 is called Gumbel-Hougaard family (they belong to Aczél-Alsina family of t–norms) and family #5 is known as Frank family (the same as the family of Frank t–norms).

Among the parameterized families of t–norms, mentioned in Table 3.1, the following are copulas: all Frank t–norms, Schweizer–Sklar t–norms with $\lambda \in$

[16] These properties are due to the fact that Archimedean copulas are continuous Archimedean t–norms.

$[-\infty, 1]$, Hamacher t–norms with $\lambda \in [0, 2]$, Yager t–norms with $\lambda \in [1, \infty]$, Dombi t–norms with $\lambda \in [1, \infty]$, Sugeno–Weber t–norms with $\lambda \in [0, \infty]$, Aczél–Alsina t–norms with $\lambda \in [1, \infty]$ and all Mayor–Torrens t–norms.

Table 3.4. Limiting and special cases of copulas in Table 3.3

#	**Limiting and special cases of copulas**
1.	$C_{-1} = T_L, \quad C_0 = T_P, \quad C_1 = \frac{\Pi}{\Sigma - \Pi}, \quad C_\infty = \min$
2.	$C_1 = T_L, \quad C_\infty = \min$
3.	$C_0 = T_P, \quad C_1 = \frac{\Pi}{\Sigma - \Pi}$
4.	$C_1 = T_P, \quad C_\infty = \min$
5.	$C_{-\infty} = T_L, \quad C_0 = T_P, \quad C_\infty = \min$

3.6 Other conjunctive and disjunctive functions

By no means t–norms and t–conorms (or their weighted counterparts) are the only conjunctive and disjunctive aggregation functions. We already gave the definition of copulas (Definition 3.108) and semi- and quasi-copulas (Definitions 3.6 and 3.8), that are conjunctive aggregation functions with neutral element $e = 1$. In this section we discuss how to construct many parameterized families of aggregation functions (of any dimension) based on a given semicopula, a monotone non-decreasing univariate function $g : [0, 1] \to [0, 1]$ and a pseudo-disjunction, defined below. This construction was proposed in [12], and we will closely follow this paper. We will only specify the results for conjunctive aggregation functions, as similar results for disjunctive functions are obtained by duality.

Definition 3.121 (Generating function). *A monotone non-decreasing function $g : [0, 1] \to [0, 1]$, $g(1) = 1$ will be called a generating function.*

Definition 3.122 (Pseudo-disjunction). *A monotone non-decreasing function $s : [0, 1]^n \to [0, 1]$, with absorbing element $a = 1$ is called a pseudo-disjunction.*

Note 3.123. A pseudo-disjunction is not always an aggregation function, as it may fail condition $s(\mathbf{0}) = 0$. For example, $s(\mathbf{x}) = 1$ for all $\mathbf{x} \in [0, 1]^n$ is a pseudo-disjunction.

Proposition 3.124. *[12] Let h_1 and h_2 be a bivariate and n-variate semicopulas respectively and let $s : [0,1]^n \to [0,1]$ be a pseudo-disjunction. The functions*

$$f_1(\mathbf{x}) = h_1(h_2(\mathbf{x}), s(\mathbf{x})),$$
$$f_2(\mathbf{x}) = h_1(s(\mathbf{x}), h_2(\mathbf{x}))$$

are also semicopulas.

Example 3.125. Let us take the basic t–norms T_L, min and T_P as semicopulas, and their dual t–conorms as pseudo-disjunctions. Then we obtain the following extended aggregation functions

$$\begin{aligned}
f_1(\mathbf{x}) &= \min(\mathbf{x}) S_L(\mathbf{x}), && (h_1 = T_P, h_2 = \min, s = S_L);\\
f_2(\mathbf{x}) &= T_P(\mathbf{x}) \max(\mathbf{x}), && (h_1 = h_2 = T_P, s = \max);\\
f_3(\mathbf{x}) &= T_P(\mathbf{x}) S_L(\mathbf{x}), && (h_1 = h_2 = T_P, s = S_L);\\
f_4(\mathbf{x}) &= \max\{\min(\mathbf{x}) + S_L(\mathbf{x}) - 1, 0\}, && (h_1 = T_L, h_2 = \min, s = S_L).
\end{aligned}$$

These aggregation functions are defined for any n (hence we call them extended aggregation functions), but they are not associative.

Let us take n generating functions $g_1, g_2, \ldots, g_n$. It is clear that if s is a disjunctive aggregation function, then so is $\tilde{s}(\mathbf{x}) = s(g_1(x_1), g_2(x_2), \ldots, g_n(x_n))$. Then we have

Corollary 3.126. *Let h_1 and h_2 be a bivariate and n-variate semicopulas respectively, s be a disjunctive aggregation function, $g_i(t), i = 1, \ldots, n$ be generating functions, and let $\tilde{s} : [0,1]^n \to [0,1]$ be defined as $\tilde{s}(\mathbf{x}) = s(g_1(x_1), g_2(x_2), \ldots, g_n(x_n))$. The functions*

$$f_1(\mathbf{x}) = h_1(h_2(\mathbf{x}), \tilde{s}(\mathbf{x}))$$
$$f_2(\mathbf{x}) = h_1(\tilde{s}(\mathbf{x}), h_2(\mathbf{x}))$$

are semicopulas.

By using parameterized families of pseudo-disjunctions and t–norms, we obtain large classes of asymmetric non-associative semicopulas, exemplified below for $n = 2$.

Example 3.127.

$$g_p(t) = \begin{cases} 1, \text{ if } t \geq p, \\ 0 \ \text{ otherwise,} \end{cases}$$

for $p \in [0,1]$. Let $h_2 = \min$, h_1 be an arbitrary semicopula, and s an arbitrary disjunctive aggregation function. Then

$$f(x,y) = h_1(\min(x,y), s(g_p(x), g_q(y))) = \begin{cases} \min(x,y), & \text{if } p \leq x \text{ or } q \leq y, \\ 0 & \text{otherwise.} \end{cases}$$

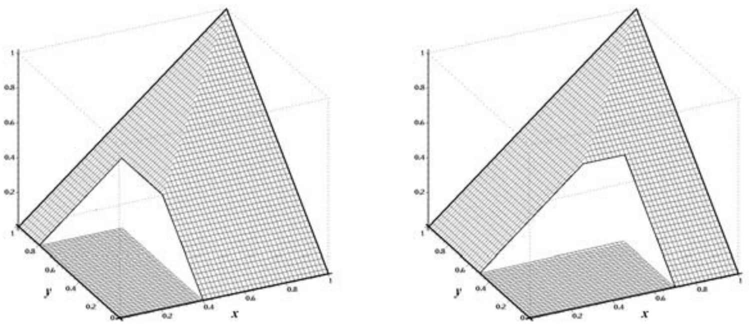


Fig. 3.19. 3D plots of semicopulas in Example 3.127 with $p = 0.4$, $q = 0.8$ (left) and $p = 0.7$, $q = 0.5$ (right).

Example 3.128. Let $g_p(t) = \max(1 - p(1 - t), 0)$, $h_2 = \min$, $h_1 = T_P$ and $s = \max$. Then

$$f(x, y) = h_1(\min(x, y), s(g_p(x), g_q(y))) = \min(x, y) \cdot \max\{1 - p(1 - x), 1 - q(1 - y), 0\}$$

Example 3.129. Let $g_p(t) = t^p$, $h_1 = h_2 = \min$, and $s = \max$. Then

$$f(x, y) = h_1(\min(x, y), s(g_p(x), g_q(y))) = \min\{\min(x, y), \max(x^p, y^q)\}.$$

- If $h_1 = T_P$ we obtain

  $$f(x, y) = \min(x, y) \cdot \max(x^p, y^q).$$

  For $p = q$ this simplifies to

  $$f(x, y) = \begin{cases} yx^p, & \text{if } x > y, \\ xy^p & \text{otherwise.} \end{cases}$$

- If $s = S_L$ we obtain $f(x, y) = \min(x, y) \cdot \min(1, x^p + y^q)$,
- If $s = S_P$ we get $f(x, y) = \min(x, y)(x^p + y^q - x^p y^q)$.

Example 3.130. Using $h_1 = h_2 = T_P$, $g_p(t) = t^p$ and various s we get

$$\begin{aligned} f(x, y) &= xy \max(x^p, y^q), \\ f(x, y) &= xy(x^p + y^q - x^p y^q), \\ f(x, y) &= xy \min(1, x^p + y^q). \end{aligned}$$

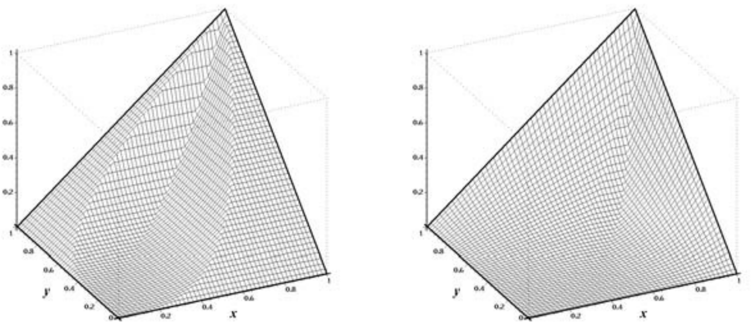


Fig. 3.20. 3D plots of semicopulas in Example 3.129 with $p = 3$, $q = 5$ and $f(x,y) = \min\{\min(x,y), \max(x^p, y^q)\}$ (left), and $p = 5$, $q = 3$ and $f(x,y) = \min(x,y) \cdot \max(x^p, y^q)$ (right).

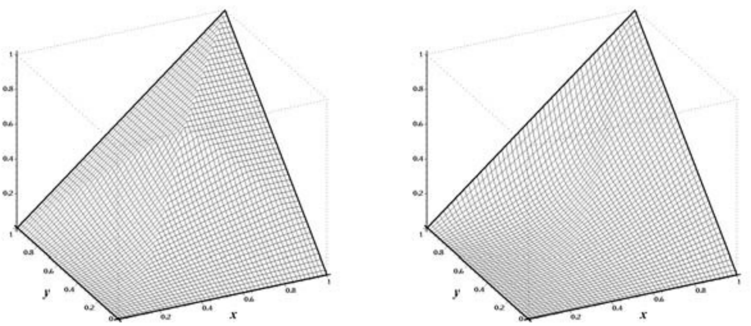


Fig. 3.21. 3D plots of semicopulas in Example 3.129 with $p = 1$, $q = 2$ and $f(x,y) = \min(x,y) \cdot \min(1, x^p + y^q)$ (left), and with $p = 2$, $q = 8$ and $f(x,y) = \min(x,y)(x^p + y^q - x^p y^q)$ (right).

Example 3.131. Several other semicopulas are proposed in [12] based on equation

$$f(\mathbf{x}) = h_1(h_2(\mathbf{x}), g(s(\mathbf{x}))),$$

for instance

$$f(x,y) = \min(x,y)(x + y - xy)^p.$$

Extension of the above examples to the n-variate case is straightforward.

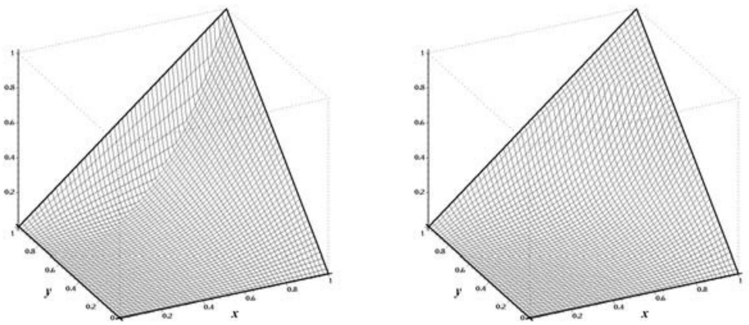


Fig. 3.22. 3D plots of semicopulas in Example 3.130 with $p = 2$, $q = 8$ and $f(x, y) = xy \max(x^p, y^q)$ (left), and with $p = 3$, $q = 5$ and $f(x, y) = xy \min(1, x^p + y^q)$ (right).

3.7 Noble reinforcement

In this section we concentrate on disjunctive aggregation functions with some special properties, because of the application they come from. By duality, equivalent results are easily obtained for conjunctive functions.

As we know, disjunctive aggregation results in mutual reinforcement of inputs. However, in some cases such a reinforcement has to be limited, and the standard aggregation functions we considered so far (notably t–conorms) are not suitable. Consider aggregation of inputs in *recommender systems*, which are frequently used in e–commerce. Recommender systems recommend customers various products based on information collected from customers (their explicit preferences or preferences deducted from their purchase history). A number of available products (alternatives) are ranked by using several criteria (e.g., whether a customer likes mystery movies, comedies, etc.). This way one has a vector of inputs $\mathbf{x}$, in which each component $x_i \in [0, 1]$ denotes the degree of satisfaction of the i-th criterion. Such degrees are called *justifications*. Each justification by itself is sufficient to recommend a product, but more than one justification provides a stronger recommendation. The products are shortlisted and displayed in the order of the degree of recommendation.

In our terminology, we have an aggregation function f which combines justifications, with the properties:

- f is continuous;
- f has neutral element $e = 0$ (and thus f is disjunctive);
- f is symmetric.

We need f to be defined for any number of arguments, but we do not require associativity. It appears that any triangular conorm S will be suitable as an aggregation function. However, when we evaluate $S(\mathbf{x})$ for some typical

vectors of inputs, we see that t–conorms have an undesirable *tail effect*: aggregation of several small inputs results in a very strong recommendation. For example, take the Łukasiewicz t–conorm, the inputs $\mathbf{x} = (0.1, 0.1, \ldots, 0.1)$ and $n = 10$.

$$S_L(\mathbf{x}) = \min\{1, \sum_{i=1}^{10} 0.1\} = 1,$$

i.e., we obtain the strongest recommendation. Similarly, take $\mathbf{x} = (0.3, \ldots, 0.3)$ and the probabilistic sum t–conorm

$$S_P(\mathbf{x}) = 1 - \prod_{i=1}^{10}(1 - 0.3) = 0.97,$$

also a very strong recommendation. In fact, for all parametric families of t–conorms we have considered so far (except the maximum t–conorm), we have a similar effect: several weak justifications reinforce each other to give a very strong recommendation. In the context of recommender systems it is undesirable: it is not intuitive to recommend a product with several very weak justifications, and moreover, rank it higher than a product with just one or two strong justifications. A customer is more likely to purchase a product which strongly matches one-two criteria than a product which does not match well any of the criteria.

In contrast, the maximum t–conorm does not provide any reinforcement: the value of the recommendation is that of the strongest justification only. This is also undesirable.

R. Yager [270] proposed the concept of *noble* reinforcement, where only strong inputs reinforce each other, while weak inputs do not. In what follows, we define the thresholds for weak and strong justifications (which are in fact labels of fuzzy sets), and then discuss prototypical situations, in which reinforcement of inputs needs to be limited. These include the number of justifications, the number of *independent* justifications, the number of strong and weak justifications and also various combinations.

In its simplest form, the noble reinforcement requirement can be defined verbally as follows.

Noble reinforcement requirement 1 "If some justifications highly recommend an object, without completely recommending it, we desire to allow these strongly supporting scores to reinforce each other."

To express this requirement mathematically, we will first define a crisp threshold α to characterize *high* values. Let $\alpha \in [0, 1]$, and the interval $[\alpha, 1]$ be the set of high input values. Later we will fuzzify this interval by using TSK methodology [231], but at the moment we concentrate on crisp intervals. Let also E denote a subset of indices $\mathcal{E} \subseteq \{1, \ldots, n\}$ and $\tilde{\mathcal{E}}$ denote its complement.

Definition 3.132. *An extended aggregation function F_α has a noble reinforcement property with respect to a crisp threshold $\alpha \in [0, 1]$ if it can be expressed as*

$$F_\alpha(\mathbf{x}) = \begin{cases} A_{i\in E}(\mathbf{x}), & \textit{if } \exists \mathcal{E} \subseteq \{1,\ldots,n\} | \forall i \in \mathcal{E} : x_i \geq \alpha \\ & \textit{and } \forall i \in \tilde{\mathcal{E}} : x_i < \alpha, \\ \max(\mathbf{x}), & \textit{otherwise,} \end{cases} \tag{3.24}$$

where $A_{i\in\mathcal{E}}(\mathbf{x})$ is a disjunctive extended aggregation function (i.e., greater than or equal to maximum), applied only to the components of $\mathbf{x}$ with the indices in $\mathcal{E}$.

That is, only the components of $\mathbf{x}$ greater than α are reinforced. This definition immediately implies that no continuous Archimedean t–conorm has the noble reinforcement property (this follows from Proposition 3.35). Therefore we shall use the ordinal sum construction.

Proposition 3.133. *[26] Let S be a t-conorm and let $\alpha \in [0,1]$. Define a triangular conorm by means of an ordinal sum ($< \alpha, 1, S >$), or explicitly,*

$$S_\alpha(x_1,x_2) = \begin{cases} \alpha + (1-\alpha)S(\frac{x_1-\alpha}{1-\alpha}, \frac{x_2-\alpha}{1-\alpha}), & \textit{if } x_1, x_2 \geq \alpha, \\ \max(x_1,x_2), & \textit{otherwise,} \end{cases} \tag{3.25}$$

where $S \neq \max$; $S_\alpha(x_1,\ldots,x_n)$ (defined by using associativity for any dimension n) has the noble reinforcement property.

Note 3.134. A special case of the t–conorm in Proposition 3.133 is the dual of a Dubois-Prade t-norm (see [142, 286]), expressed as $(< 0, \alpha, T_P >)$.

Proposition 3.133 gives a generic solution to the noble reinforcement problem, defined (through associativity) for any number of arguments. Next we discuss refinements of the noble reinforcement requirement, which involve not only the threshold for high values, but also the minimal number of inputs to be reinforced, their independence, and also the presence of low input values, [26, 270]. In many systems, notably the recommender systems, some criteria may not be independent, e.g., when various justifications measure essentially the same concept. It is clear that mutual reinforcement of correlated criteria should be smaller than reinforcement of truly independent criteria.

First we specify the correspondent requirements in words and then give their mathematical definitions.

Requirement 2 Provide reinforcement if at least k arguments are *high.*

Requirement 3 Provide reinforcement of at least k *high* scores, if at least m of these scores are *very high.*

Requirement 4 Provide reinforcement of at least k *high* scores, if we have *at most m* low scores.

Requirement 5 Provide reinforcement of at least $k > 1$ *independent high* scores.

All these requirements are formulated using fuzzy sets of *high, very high* and *low* scores, and also a fuzzy set of *independent high* scores. We shall

formulate the problem first using crisp sets, and then fuzzify them using TSK (Takagi-Sugeno-Kang) methodology [231].

Define three crisp thresholds α, β, γ, $\gamma < \alpha < \beta$; the interval $[\alpha, 1]$ will denote *high* scores and the interval $[\beta, 1]$ will denote *very high* scores, and the interval $[0, \gamma]$ will denote *low* scores.

Translating the above requirements into mathematical terms, we obtain:

Definition 3.135. *An extended aggregation function $F_{\alpha,k}$ provides noble reinforcement of at least k arguments with respect to a crisp threshold $\alpha \in [0,1]$, if it can be expressed as*

$$F_{\alpha,k}(\mathbf{x}) = \begin{cases} A_{i\in\mathcal{E}}(\mathbf{x}), & \text{if } \exists\mathcal{E} \subseteq \{1,\dots,n\}|\ \ |\mathcal{E}| \geq k, \\ & \forall i \in \mathcal{E} : x_i \geq \alpha, \\ & \text{and } \forall i \in \tilde{\mathcal{E}} : x_i < \alpha, \\ \max(\mathbf{x}), & \text{otherwise}, \end{cases} \tag{3.26}$$

where $A_{i\in\mathcal{E}}(\mathbf{x})$ is a disjunctive extended aggregation function, applied to the components of $\mathbf{x}$ with the indices in $\mathcal{E}$.

Definition 3.136. *An extended aggregation function $F_{\alpha,\beta,k,m}$ provides noble reinforcement of at least k high values, with at least m very high values, with respect to thresholds $\alpha, \beta \in [0,1]$, $\alpha < \beta$ if it can be expressed as*

$$F_{\alpha,\beta,k,m}(\mathbf{x}) = \begin{cases} A_{i\in\mathcal{E}}(\mathbf{x}), & \text{if } \exists\mathcal{E} \subseteq \{1,\dots,n\}|\ |\mathcal{E}| \geq k, \\ & \forall i \in \mathcal{E} : x_i \geq \alpha, \forall i \in \tilde{\mathcal{E}} : x_i < \alpha, \\ & \text{and } \exists\mathcal{D} \subseteq \mathcal{E}|\ |\mathcal{D}| = m, \\ & \forall i \in \mathcal{D} : x_i > \beta, \\ \max(\mathbf{x}), & \text{otherwise}, \end{cases} \tag{3.27}$$

where $A_{i\in\mathcal{E}}(\mathbf{x})$ is any disjunctive extended aggregation function, applied only to the components of $\mathbf{x}$ with the indices in $\mathcal{E}$.

Definition 3.137. *An extended aggregation function $F_{\alpha,\gamma,k,m}$ provides noble reinforcement of at least k high values, with at most m low values, with respect to thresholds $\alpha, \gamma \in [0,1]$, $\gamma < \alpha$ if it can be expressed as*

$$F_{\alpha,\gamma,k,m}(\mathbf{x}) = \begin{cases} A_{i\in\mathcal{E}}(\mathbf{x}), & \text{if } \exists\mathcal{E} \subseteq \{1,\dots,n\}|\ |\mathcal{E}| \geq k, \\ & \forall i \in \mathcal{E} : x_i \geq \alpha, \forall i \in \tilde{\mathcal{E}} : x_i < \alpha, \\ & \text{and } \exists\mathcal{D} \subseteq \{1,\dots,n\}| \\ & |\mathcal{D}| = n - m, \forall i \in \mathcal{D} : x_i \geq \gamma, \\ \max(\mathbf{x}), & \text{otherwise}, \end{cases} \tag{3.28}$$

where $A_{i\in\mathcal{E}}(\mathbf{x})$ is any disjunctive extended aggregation function, applied only to the components of $\mathbf{x}$ with the indices in $\mathcal{E}$.

That is, we desire to have reinforcement when the scores are high or medium, and explicitly prohibit reinforcement if some of the scores are low. In the above, $1 < k \leq n$ and $0 \leq m \leq n-k$; when $m = 0$ we prohibit reinforcement when at least one low score is present.

Construction of the extended aggregation functions using Definitions 3.135-3.137 seems to be straightforward (we only need to choose an appropriate disjunctive function $A_{i \in \mathcal{E}}$), however at closer examination, it leads to discontinuous functions. For applications we would like to have continuity, moreover, even Lipschitz continuity, as it would guarantee stability of the outputs with respect to input inaccuracies. A general method of such construction is presented in Chapter 6, and it involves methods of monotone Lipschitz interpolation. We will postpone the technical discussion till Chapter 6, and at the moment assume that suitable extended aggregation functions $F_{\alpha,k}, F_{\alpha,\beta,k,m}, F_{\alpha,\gamma,k,m}$ are given.

Fuzzification

Next we fuzzify the intervals of *high, very high* and *low* values. Define a fuzzy set *high* by means of a membership function $\mu_h(t)$. Note that $\mu_h(t)$ is a strictly increasing bijection of $[0,1]$. Similarly, we define the fuzzy set *very high* by means of a membership function $\mu_{vh}(t)$, also a strictly increasing bijection of $[0,1]$, and the fuzzy set *low* by means of a strictly decreasing bijection $\mu_l(t)$. Let us also define a membership function $\mu_q(j)$ to denote membership values in the fuzzy set *the minimal number of components*. The higher the number of *high* components of $\mathbf{x}$, starting from some minimal number, the stronger reinforcement. Let us also denote $\min(\mathcal{E}) = \min_{i \in \mathcal{E}} x_i$

Yager calculates the value of the extended aggregation function with noble reinforcement (based on Definition 3.132) using

$$\begin{aligned} F(\mathbf{x}) &= \max_{\mathcal{E}}\{\mu_h(\min(\mathcal{E}))A_{i \in \mathcal{E}}(\mathbf{x}) + (1 - \mu_h(\min(\mathcal{E}))) \max(\mathbf{x})\} \\ &= \max_{\mathcal{E}}\{\max(\mathbf{x}) + \mu_h(\min(\mathcal{E}))(A_{i \in \mathcal{E}}(\mathbf{x}) - \max(\mathbf{x}))\}. \end{aligned} \tag{3.29}$$

Similarly, based on Definition 3.135 we obtain

$$F(\mathbf{x}) = \max_{\mathcal{E},k}\{\max(\mathbf{x}) + \mu_h(\min(\mathcal{E}))\mu_q(k)(F_{\alpha,k}(\mathbf{x}) - \max(\mathbf{x}))\}, \tag{3.30}$$

where $F_{\alpha,k}(\mathbf{x})$ is computed for fixed k and $\mathcal{E}$. Based on Definition 3.136 we obtain

$$F(\mathbf{x}) = \max_{\mathcal{E},\mathcal{D}}\{\max(\mathbf{x}) + \mu_h(\min(\mathcal{E}))\mu_{vh}(\min(\mathcal{D}))(F_{\alpha,\beta,k,m}(\mathbf{x}) - \max(\mathbf{x}))\}. \tag{3.31}$$

Based on Definition 3.137 we obtain

$$F(\mathbf{x}) = \max_{\mathcal{E},\mathcal{D}}\{\max(\mathbf{x}) + \mu_h(\min(\mathcal{E}))(1 - \mu_l(\min(\mathcal{D})))(F_{\alpha,\gamma,k,m}(\mathbf{x}) - \max(\mathbf{x}))\}. \tag{3.32}$$

As far as the requirement of independent scores is concerned, we need to define a function which measures the degree of independence of the criteria. We shall use a monotone non-increasing function defined on the subsets of criteria $\mu_I : 2^{\mathcal{N}} \to [0,1]$, ($\mathcal{N} = \{1, \ldots, n\}$) such that the value $\mu_I(\mathcal{E})$ represents the degree of mutual independence of the criteria in the set $\mathcal{E} \subseteq \{1, 2, \ldots, n\}$. Note that having a larger subset does not increase the degree of independence $\mathcal{A} \subseteq \mathcal{B} \implies \mu_I(\mathcal{A}) \geq \mu_I(\mathcal{B})$. We assume that the function $\mu_I(\mathcal{E})$ is given.

Example 3.138. Consider a simplified recommender system for online movie sales, which recommends movies based on the following criteria. The user likes movies: 1) mystery, 2) detectives, 3) drama, and 4) science fiction. One could use the following membership function $\mu_I(\mathcal{E})$

$$\begin{aligned}
\mu_I(\{i\}) &= 1, i = 1, \ldots, 4; \\
\mu_I(\{1,2\}) &= \mu_I(\{1,2,3\}) = \mu_I(\{1,2,4\}) = 0.7; \\
\mu_I(\{1,3\}) &= \mu_I(\{1,4\}) = \mu_I(\{2,3\}) = \mu_I(\{2,4\}) = 1; \\
\mu_I(\{3,4\}) &= \mu_I(\{2,3,4\}) = \mu_I(\{1,3,4\}) = \mu_I(\{1,2,3,4\}) = 0.5.
\end{aligned}$$

We obtain an aggregation function which satisfies Requirement 5 by taking the maximum over all possible subsets of criteria $\mathcal{E}$ in which reinforcement takes place,

$$F(\mathbf{x}) = \max_{\mathcal{E},k} \left(\max(\mathbf{x}) + \ \mu_h(\min(\mathcal{E}))\mu_q(k)\mu_I(\mathcal{E})(A_{i\in\mathcal{E}}(\mathbf{x}) - \max(\mathbf{x})) \right), \quad (3.33)$$

where $A_{i\in\mathcal{E}}$ is computed as in (3.24) but only with a fixed subset $\mathcal{E}$.

Note 3.139. Note that the maximum over all $\mathcal{E}$ is essential: because the function μ_I is non-increasing, the function $F(\mathbf{x})$ may fail to be monotone. For example, consider the problem with three linearly dependent, but pairwise independent criteria, $\mu(\{1,2\}) = \mu(\{2,3\}) = \mu(\{1,3\}) = 1$, $\mu(\{1,2,3\}) = 0$. Now if all components of $\mathbf{x}$ are high we have no reinforcement. But if two components are high, and the remaining is small, we have reinforcement. We ensure monotonicity by taking maximum reinforcement over all possible combinations of the criteria.

3.8 Key references

C. Alsina, M.J. Frank, and B. Schweizer. *Associative Functions: Triangular Norms and Copulas.* World Scientific, Singapore, 2006.

T. Calvo, A. Kolesárová, M. Komorníková, and R. Mesiar. Aggregation operators: properties, classes and construction methods. In T. Calvo, G. Mayor, and R. Mesiar, editors, *Aggregation Operators. New Trends and Applications*, pages 3–104. Physica-Verlag, Heidelberg, New York, 2002.

D. Dubois and H. Prade. *Fuzzy Sets and Systems. Theory and Applications.* Academic Press, New York, 1980.

E.P. Klement, R. Mesiar, and E. Pap. *Triangular Norms.* Kluwer, Dordrecht, 2000.

R.B. Nelsen. *An Introduction to Copulas.* Springer, Berlin, 1998.

B. Schweizer and A. Sklar. *Probabilistic Metric Spaces.* North-Holland, New York, 1983.

4

Mixed Functions

4.1 Semantics

Mixed aggregation functions are those whose behavior depends on the inputs. These functions exhibit conjunctive, disjunctive or averaging behavior on different parts of their domain. We have the following general definition.

Definition 4.1 (Mixed aggregation). *An aggregation function is mixed if it is neither conjunctive, nor disjunctive or averaging, i.e., it exhibits different types of behavior on different parts of the domain.*

Note 4.2. An immediate consequence of the above definition is that mixed aggregation functions are not comparable with min and/or are not comparable with max.

The main use of mixed aggregation functions is in those situations where some inputs positively reinforce each other, while other inputs have negative or no reinforcement. This is strongly related to bipolar aggregation, in which some inputs are considered as "positive" and others as "negative" evidence, see Section 1.5. For example, in expert systems such as MYCIN and PROSPECTOR [38, 88], certain pieces of evidence confirm a hypothesis, whereas others disconfirm it (these are called *certainty factors*). This is modeled by positive and negative inputs on the scale $[-1, 1]$, with 0 being a "neutral" value. We know, however, that any bounded scale can be transformed into $[0, 1]$ (see p. 31), therefore we shall use the inputs from the unit interval, with $\frac{1}{2}$ being the "neutral" value, and interpret the inputs smaller than $\frac{1}{2}$ as "negative" evidence and those larger than $\frac{1}{2}$ as "positive" evidence.

Example 4.3. Consider the following rule system
Symptom A confirms diagnosis D (with certainty α);
Symptom B disconfirms diagnosis D (with certainty β);
Symptom C confirms diagnosis D (with certainty γ);
etc.

The inputs are: A (with certainty a), B (with certainty b), C (with certainty c).

The certainty of the diagnosis D on $[-1,1]$ scale is calculated using $f(g(a,\alpha), -g(b,\beta), g(c,\gamma))$, where f is conjunctive for negative inputs, disjunctive for positive inputs and averaging elsewhere, and g is a different conjunctive aggregation function (e.g., $g = \min$).

A different situation in which the aggregation function needs to be of mixed type is when modeling heterogeneous rules like

$$\textbf{If } t_1 \textit{ is } A_1 \textbf{ AND } (t_2 \textit{ is } A_2 \textbf{ OR } t_3 \textit{ is } A_3) \textbf{ THEN ...}$$

and $x_1, x_2, \ldots$ denote the degrees of satisfaction of the rule predicates t_1 is A_1, t_2 is A_2, etc. Here we want to aggregate the inputs $x_1, x_2, \ldots$ using a single function $f(\mathbf{x})$, but the rule is a mixture of conjunction and disjunction.

Consider two specific examples of mixed type aggregation functions.

Example 4.4. MYCIN [38] is the name of a famous expert system which was one of the first systems capable of reasoning under uncertainty. Certainty factors, represented by numbers on the bipolar scale $[-1,1]$, were combined by means of the function

$$f(x,y) = \begin{cases} x + y - xy, & \text{if } \min(x,y) \geq 0, \\ \frac{x+y}{1-\min(|x|,|y|)}, & \text{if } \min(x,y) < 0 < \max(x,y), \\ x + y + xy, & \text{if } \max(x,y) \leq 0. \end{cases} \tag{4.1}$$

This function is symmetric and associative (recall that the latter implies that it is defined uniquely for any number of arguments), but does not define outputs at $(-1,1)$ and $(1,-1)$. It is understood though that the output is -1 in these cases.

On $[0,1]$ scale it is given as

$$f(x,y) = \begin{cases} 2(x + y - xy) - 1, & \text{if } \min(x,y) \geq \frac{1}{2}, \\ \frac{x+y-1}{1-\min(|2x-1|,|2y-1|)} + \frac{1}{2}, & \text{if } \min(x,y) < \frac{1}{2} < \max(x,y), \\ 2xy, & \text{if } \max(x,y) \leq \frac{1}{2}. \end{cases} \tag{4.2}$$

Example 4.5. PROSPECTOR was another pioneering expert system for mineral exploration [88]. PROSPECTOR's aggregation function on the scale $[-1,1]$ is defined as

$$f(x,y) = \frac{x+y}{1+xy}. \tag{4.3}$$

It is symmetric and associative. It is understood that $f(-1,1) = -1$.

On $[0,1]$ scale it is given as

$$f(x,y) = \frac{x+y}{xy + (1-x)(1-y)}. \tag{4.4}$$

In both mentioned examples, the aggregation functions exhibited conjunctive behavior on $[0, \frac{1}{2}]^n$ and disjunctive behavior on $[\frac{1}{2}, 1]^n$. On the rest of the domain the behavior was averaging. This is not the only way to partition the domain into disjunctive, conjunctive and averaging parts, as we shall see later in this Chapter.

The class of mixed aggregation functions includes many different families. Some of them, such as *uninorms, nullnorms*, compensatory *T-S functions* and *ST-OWAs*, are related – in some sense which will be detailed later – to triangular norms and conorms. Other important families of mixed functions are the *symmetric sums* and some kinds of *generated functions.*

Uninorms and nullnorms are two popular families of associative aggregation functions with clearly defined behavior (see Figure 4.1 for the two-dimensional case):

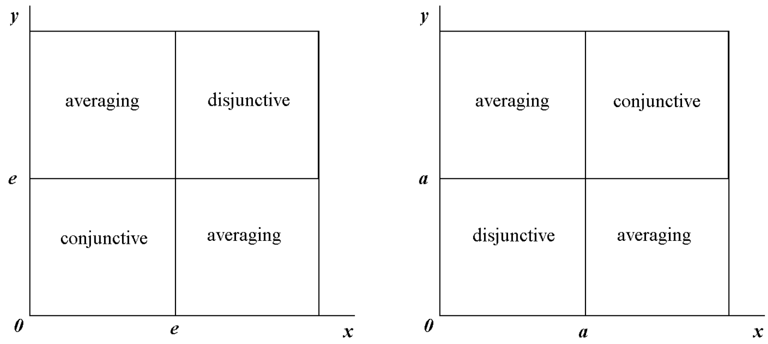


Fig. 4.1. Behavior of uninorms (left) and nullnorms (right).

- *Uninorms* are associative aggregation functions that present conjunctive behavior when dealing with low input values (those below a given value e which is, in addition, the neutral element), have disjunctive behavior for high values (those above e) and are averaging otherwise (i.e., when receiving a mixture of low and high inputs).
- On the other hand, *nullnorms* are associative aggregation functions that are disjunctive for low values (those below a given value a which is, in addition, the absorbing element), conjunctive for high values (those above a) and are otherwise averaging (actually, in these cases they provide as the output the absorbing element a).

The aggregation functions in Examples 4.4 and 4.5 turn out to be uninorms, as we shall see in the next section. Of course, not only uninorms and nullnorms exhibit the behavior illustrated on Fig. 4.1: some generated functions discussed in Section 4.4 and some symmetric sums behave as uninorms

(but they are not associative). However, uninorms and nullnorms are the only types of *associative* functions with the behavior on Fig. 4.1, so we start with them.

4.2 Uninorms

Uninorms[1] were introduced as a generalization of t-norms and t-conorms based on the observation that these two classes of aggregation functions (see Chapter 3) are defined by means of the same three axioms – associativity, symmetry and possession of a neutral element – just differing in the value of the latter, which is 1 for t-norms and 0 for t-conorms. This observation leads to studies of associative and symmetric aggregation functions that have a neutral element which may take any value between the two end points of the unit interval.

Since their introduction, uninorms have proved to be useful for practical purposes in different situations, and have been applied in different fields[2].

4.2.1 Definition

Uninorms are defined in the bivariate case as follows:

Definition 4.6 (Uninorm). *A uninorm is a bivariate aggregation function* $U : [0,1]^2 \to [0,1]$ *which is associative, symmetric and has a neutral element* e *belonging to the open interval* $]0,1[$.

Note 4.7. An alternative definition allows the neutral element e to range over the whole interval $[0,1]$, and thus includes t–norms and t–conorms as special limiting cases.

Note 4.8. Since uninorms are associative, they are extended in a unique way to functions with any number of arguments. Thus uninorms constitute a class of extended aggregation functions (see Definition 1.6 on p. 4).

4.2.2 Main properties

The general behavior of uninorms is depicted in Figure 4.1, which shows that a uninorm with neutral element e is conjunctive in the square $[0,e]^2$ and disjunctive in the square $[e,1]^2$. More precisely, uninorms act as t-norms in $[0,e]^2$ and as t-conorms in $[e,1]^2$, that is, any uninorm with neutral element e is associated with a t-norm T_U and a t-conorm S_U such that:

[1] Uninorms appeared under this name in 1996 [278], but a special class of them – nowadays known as the class of representable uninorms – was first studied in 1982 [78].

[2] For example, expert systems [65, 246] or fuzzy systems modeling [269, 277].

$$\forall (x,y) \in [0,e]^2, \qquad U(x,y) = e \cdot T_U\left(\frac{x}{e}, \frac{y}{e}\right),$$
$$\forall (x,y) \in [e,1]^2, \qquad U(x,y) = e + (1-e) \cdot S_U\left(\frac{x-e}{1-e}, \frac{y-e}{1-e}\right).$$

The functions T_U and S_U are usually referred to as the underlying t-norm and t-conorm related to the uninorm U.

On the remaining parts of the unit square, uninorms have averaging behavior, i.e:

$$\forall (x,y) \in [0,e] \times [e,1] \cup [e,1] \times [0,e], \qquad \min(x,y) \le U(x,y) \le \max(x,y).$$

Note that contrary to what happens in the squares $[0,e]^2$ and $[e,1]^2$, the behavior of uninorms in the rest of the unit square is not tied to any specific class of (averaging) functions.

The structure of uninorms that has just been described is summarized in Figure 4.2, using the following notation:

$$T_U^*(x,y) = e \cdot T_U\left(\frac{x}{e}, \frac{y}{e}\right),$$
$$S_U^*(x,y) = e + (1-e) \cdot S_U\left(\frac{x-e}{1-e}, \frac{y-e}{1-e}\right).$$

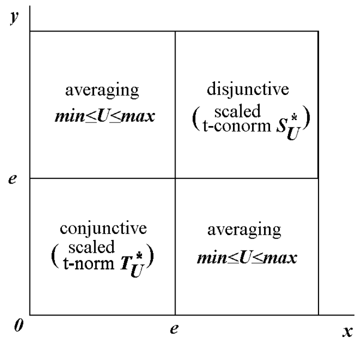


Fig. 4.2. Structure of a uninorm with neutral element e, underlying t-norm T_U and underlying t-conorm S_U.

Other interesting properties of uninorms are the following:

Absorbing element For any uninorm $U(0,1) \in \{0,1\}$. This allows one to classify uninorms into two different categories:

- *Conjunctive uninorms* are uninorms which verify $U(0,1) = 0$. These uninorms have absorbing element $a = 0$ (due to monotonicity).

- *Disjunctive uninorms* are uninorms which verify $U(0,1) = 1$. These uninorms have absorbing element $a = 1$ (due to monotonicity).

Duality The class of uninorms is closed under duality, that is, the dual of any uninorm U with respect to an arbitrary strong negation N, defined as

$$U_d(x,y) = N(U(N(x), N(y))),$$

is also a uninorm, and it has the following properties:

- If U has neutral element e, U_d has neutral element $N(e)$.
- If U is a conjunctive (respectively disjunctive) uninorm, then U_d is a disjunctive (resp. conjunctive) uninorm.

Clearly, no uninorm can be self-dual, since at least the values $U(0,1)$ and $U_d(0,1) = N(U(1,0)) = N(U(0,1))$ will be different.

Continuity Uninorms are never continuous on the whole unit square. Nevertheless, it is possible to find uninorms that are continuous on the open square $]0,1[^2$. Moreover, there are uninorms which are *almost continuous*, i.e., which are continuous everywhere except at the corners $(0,1)$ and $(1,0)$. These are called *representable uninorms*, see Section 4.2.3.

Idempotency Recall from Chapter 3, that the only idempotent t-norms and t-conorms are minimum and maximum respectively. In the case of uninorms, there are different kinds of idempotent functions[3] that have been found and characterized [64, 170].

Strict monotonicity The existence of an absorbing element prevents uninorms from being strictly monotone in the whole unit square. Notwithstanding, some of them (such as the already mentioned representable uninorms) are strictly monotone on the open square $]0,1[^2$.

4.2.3 Main classes of uninorms

There are several classes of uninorms that have been identified and characterized. Two of the most important and useful ones are described below.

The families $\mathcal{U}_{\min}$ and $\mathcal{U}_{\max}$

In Section 4.2.2 we saw that in the region $[0,e] \times [e,1] \cup [e,1] \times [0,e]$ uninorms have averaging behavior. This raises the question of whether it is possible to have uninorms acting on this region exactly as the limiting averaging functions min and max. The answer is affirmative; this idea provides two important families of conjunctive and disjunctive uninorms, known, respectively, as $\mathcal{U}_{\min}$ and $\mathcal{U}_{\max}$ (see figure 4.3).

Proposition 4.9 (Uninorms $\mathcal{U}_{\min}$ and $\mathcal{U}_{\max}$).

[3] Of course, because of monotonicity (see p. 9) such idempotent uninorms are, actually, averaging functions instead of mixed ones.

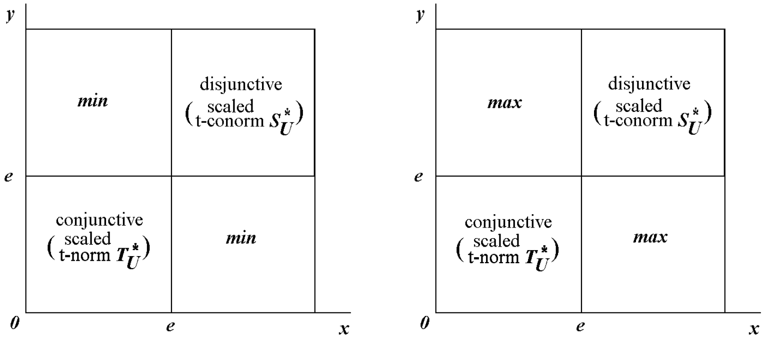


Fig. 4.3. Structure of the families of uninorms $\mathcal{U}_{\min}$ (left) and $\mathcal{U}_{\max}$ (right) as defined on p. 202.

- *Let T be an arbitrary t-norm, S be an arbitrary t-conorm and $e \in]0,1[$. The function*

$$U_{\min(T,S,e)}(x,y) = \begin{cases} e \cdot T\left(\frac{x}{e}, \frac{y}{e}\right), & \textit{if } (x,y) \in [0,e]^2, \\ e + (1-e) \cdot S\left(\frac{x-e}{1-e}, \frac{y-e}{1-e}\right), & \textit{if } (x,y) \in [e,1]^2, \\ \min(x,y) & \textit{otherwise} \end{cases}$$

  is a conjunctive uninorm with the neutral element e, and the family of all such uninorms is denoted by $\mathcal{U}_{\min}$.
- *Let T be an arbitrary t-norm, S be an arbitrary t-conorm and $e \in]0,1[$. The function*

$$U_{\max(T,S,e)}(x,y) = \begin{cases} e \cdot T\left(\frac{x}{e}, \frac{y}{e}\right), & \textit{if } (x,y) \in [0,e]^2, \\ e + (1-e) \cdot S\left(\frac{x-e}{1-e}, \frac{y-e}{1-e}\right), & \textit{if } (x,y) \in [e,1]^2, \\ \max(x,y) & \textit{otherwise} \end{cases}$$

  is a disjunctive uninorm with the neutral element e, and the family of all such uninorms is denoted by $\mathcal{U}_{\max}$.

Observe that uninorms in the two mentioned families satisfy the following properties:

- If $U \in \mathcal{U}_{\min}$ then the section U_1, given by $t \mapsto U(1,t)$, is continuous on $[0, e[$.
- If $U \in \mathcal{U}_{\max}$ then the section U_0, given by $t \mapsto U(0,t)$, is continuous on $]e, 1]$.

Note 4.10. The associativity allows one to determine uniquely the n-ary extensions of the families $\mathcal{U}_{\min}$ and $\mathcal{U}_{\max}$, which are the following:

$$U_{\min(T,S,e)}(x_1,\ldots,x_n) = \begin{cases} e \cdot T\left(\frac{x_1}{e}, \ldots, \frac{x_n}{e}\right), & \text{if } (x_1,\ldots,x_n) \in [0,e]^n, \\ e + (1-e) \cdot S\left(\frac{x_1-e}{1-e}, \ldots, \frac{x_n-e}{1-e}\right), & \text{if } (x_1,\ldots,x_n) \in [e,1]^n, \\ \min(x_1,\ldots,x_n) & \text{otherwise,} \end{cases}$$

$$U_{\max(T,S,e)}(x_1,\ldots,x_n) = \begin{cases} e \cdot T\left(\frac{x_1}{e},\ldots,\frac{x_n}{e}\right), & \text{if } (x_1,\ldots,x_n) \in [0,e]^n, \\ e + (1-e)\cdot S\left(\frac{x_1-e}{1-e},\ldots,\frac{x_n-e}{1-e}\right), & \text{if } (x_1,\ldots,x_n) \in [e,1]^n, \\ \max(x_1,\ldots,x_n) & \text{otherwise.} \end{cases}$$

Representable uninorms

We saw in Chapter 3, that there are t-norms and t-conorms — in particular, the continuous Archimedean ones, — that may be represented in terms of some specific single-variable functions known as their additive (or multiplicative) generators. A similar property exists for uninorms, i.e., there is a class of uninorms (including both conjunctive and disjunctive ones), usually known as *representable uninorms*[4], that can be built by means of univariate generators:

Definition 4.11 (Representable uninorms). *Let* $u : [0,1] \to [-\infty,+\infty]$ *be a strictly increasing bijection*[5] *such that* $u(e) = 0$ *for some* $e \in]0,1[$.

- *The function given by*

$$U(x,y) = \begin{cases} u^{-1}(u(x)+u(y)), & \textit{if } (x,y) \in [0,1]^2 \setminus \{(0,1),(1,0)\}, \\ 0 & \textit{otherwise} \end{cases}$$

  is a conjunctive uninorm with the neutral element e*, known as a* conjunctive representable uninorm*.*
- *The function given by*

$$U(x,y) = \begin{cases} u^{-1}(u(x)+u(y)), & \textit{if } (x,y) \in [0,1]^2 \setminus \{(0,1),(1,0)\}, \\ 1 & \textit{otherwise} \end{cases}$$

  is a disjunctive uninorm with the neutral element e*, known as a* disjunctive representable uninorm*.*

The function u *is called an* additive generator *of the uninorm* U *and it is determined up to a positive multiplicative constant*[6]*.*

Observe that each u provides two different uninorms, a conjunctive one and a disjunctive one, that differ only in the corners $(0,1)$ and $(1,0)$.

Note 4.12. Similarly to the case of t-norms and t-conorms, representable uninorms may also be described by means of multiplicative generators[7].

---

[4] Also known as *generated uninorms*, compare with Section 4.4.

[5] Note that such a function will verify $u(0) = -\infty$ and $u(1) = +\infty$.

[6] That is, if u as an additive generator of U, then so is $v(t) = cu(t), c > 0$.

[7] See, for example, [141], where representable uninorms are called "associative compensatory operators".

Note 4.13. In the case of n arguments, representable uninorms can be built from an additive generator u as follows:

- For inputs not containing simultaneously the values 0 and 1, that is, for tuples $(x_1, \ldots, x_n)$ belonging to $[0,1]^n \backslash \{(x_1, \ldots, x_n) : \{0,1\} \subseteq \{x_1, \ldots, x_n\}\}$:

$$U(x_1, \ldots, x_n) = u^{-1}\left(\sum_{i=1}^{n} u(x_i)\right)$$

- Otherwise — that is, for tuples containing simultaneously the values 0 and 1:
  – $U(x_1, \ldots, x_n) = 0$ for a conjunctive uninorm;
  – $U(x_1, \ldots, x_n) = 1$ for a disjunctive uninorm.

Proposition 4.14 (Properties of representable uninorms).

1. *Representable uninorms are* almost continuous *(i.e., continuous on* $[0,1]^2 \backslash \{(0,1),(1,0)\}$*). Moreover, they are the only uninorms verifying this property (a uninorm is almost continuous if and only if it is representable).*

2. *The function* $N_u : [0,1] \to [0,1]$ *given by* $N_u(t) = u^{-1}(-u(t))$ *is a strong negation with fixed point* e*, and* U *is* self-dual*, excluding the points* $(0,1)$ *and* $(1,0)$*, with respect to* N_u*, that is:*

$$U(x,y) = N_u(U(N_u(x), N_u(y))) \qquad \forall (x,y) \in [0,1]^2 \backslash \{(0,1),(1,0)\}.$$

3. *Representable uninorms verify thefollowing equalities and inequalities (note that the latter may be understood as a generalization of the Archimedean property of continuous t-norms and t-conorms):*
   a) $\forall t \in [0,1[\quad : \quad U(t,0) = 0;$
   b) $\forall t \in]0,1] \quad : \quad U(t,1) = 1;$
   c) $\forall t \in]0,1[\quad : \quad U(t, N_u(t)) = e;$
   d) $\forall t \in]0,e[\quad : \quad U(t,t) < t;$
   e) $\forall t \in]e,1[\quad : \quad U(t,t) > t.$
4. *Since their generators* u *are strictly increasing, representable uninorms are* strictly increasing *on* $]0,1[^2$.
5. *Strict monotonicity, along with almost continuity, ensure that the underlying t-norm and t-conorm of representable uninorms are necessarily* strict *(see section 3.4.3). Moreover:*
   - *If* U *is a representable uninorm with additive generator* u *and neutral element* e*, then the functions* $g, h : [0,1] \to [0,+\infty]$*, given by* $g(t) = -u(e \cdot t)$ *and* $h(t) = u(e + (1-e) \cdot t)$*, are additive generators, respectively, of the underlying strict t-norm* T_U *and strict t-conorm* S_U*.*
   - *Conversely, given a value* $e \in]0,1[$*, a strict t-norm* T *with an additive generator* g *and a strict t-conorm* S *with an additive generator* h*, the mapping* $u : [0,1] \to [-\infty, +\infty]$ *defined by*

$$u(t) = \begin{cases} -g\left(\frac{t}{e}\right), & \textit{if } t \in [0, e], \\ h\left(\frac{t-e}{1-e}\right), & \textit{if } t \in]e, 1], \end{cases}$$

is an additive generator of a representable uninorm with the neutral element e and underlying functions T and S.

Note 4.15. With regards to the last statement, it is interesting to point out[8] that given a value $e \in]0, 1[$, a strict t-norm T and a strict t-conorm S, the triplet (e, T, S) does not determine a unique (conjunctive or disjunctive) representable uninorm with neutral element e, but rather a *family* of them. This is due to the fact that additive generators of t-norms and t-conorms (see Proposition 3.43) are unique only up to a multiplicative positive constant, and then one can choose among the different additive generators of T and S. The choice of these generators does not affect the behavior of the corresponding uninorms on the squares $[0, e]^2$ and $[e, 1]^2$ (T and S always remain the underlying t-norm and t-conorm), but it does influence the behavior obtained on the remaining parts of the domain. For example, let g and h be additive generators of a strict t-norm T and a strict t-conorm S respectively, and take $e \in]0, 1[$. Then, for each $k > 0$, the function

$$u_k(t) = \begin{cases} -k \cdot g\left(\frac{t}{e}\right), & \text{if } t \in [0, e], \\ h\left(\frac{t-e}{1-e}\right), & \text{if } t \in]e, 1], \end{cases}$$

provides a representable uninorm U_k. All the members of the family $\{U_k\}_{k>0}$ have neutral element e, underlying t-norm T and underlying t-conorm S, but differ on the region $[0, e] \times [e, 1] \cup [e, 1] \times [0, e]$.

4.2.4 Examples

Example 4.16 (The weakest and t he strongest uninorms). Given $e \in]0, 1[$, the weakest and the strongest uninorms with neutral element e are given below (the structure and a 3D plot of these uninorms can be viewed, respectively, in Figures 4.4 and 4.5):

- The weakest uninorm with neutral element e is the conjunctive uninorm belonging to $\mathcal{U}_{\min}$ built by means of the weakest t-norm (the drastic product T_D) and the weakest t-conorm, max:

$$U_{\min(T_D, \max, e)}(x, y) = \begin{cases} 0, & \text{if } (x, y) \in [0, e[^2, \\ \max(x, y), & \text{if } (x, y) \in [e, 1]^2, \\ \min(x, y) & \text{otherwise.} \end{cases}$$

- The strongest uninorm with the neutral element e is the disjunctive uninorm belonging to $\mathcal{U}_{\max}$ built by means of the strongest t-norm, min, and the strongest t-conorm, S_D:

$$U_{\max(\min, S_D, e)}(x, y) = \begin{cases} \min(x, y), & \text{if } (x, y) \in [0, e]^2, \\ 1, & \text{if } (x, y) \in]e, 1]^2, \\ \max(x, y) & \text{otherwise.} \end{cases}$$

[8] For details on this, see [141, 184].

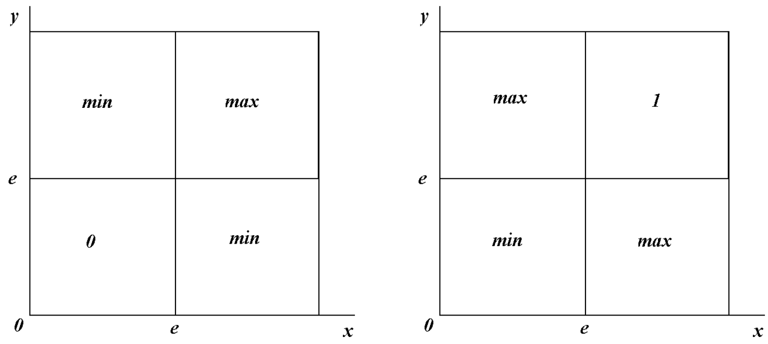


Fig. 4.4. Structure of the weakest (left) and the strongest (right) bivariate uninorms with neutral element $e \in]0, 1[$ (example 4.16, p. 206).

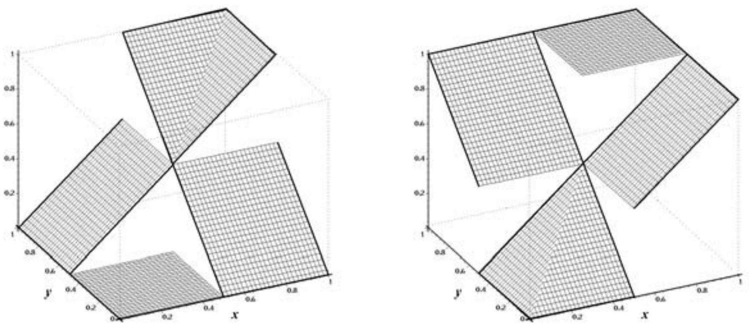


Fig. 4.5. 3D plots of the weakest (left) and the strongest (right) bivariate uninorms with neutral element $e = 0.5$ (example 4.16, p. 206).

Example 4.17 (Idempotent uninorms in $\mathcal{U}_{\min}$ and $\mathcal{U}_{\max}$). Other commonly cited examples of uninorms are the ones obtained from the families $\mathcal{U}_{\min}$ and $\mathcal{U}_{\max}$ by choosing $T = \min$ and $S = \max$ (see Figures 4.6 and 4.7):

$$U_{\min(\min,\max,e)}(x,y) = \begin{cases} \max(x,y), & \text{if } (x,y) \in [e,1]^2, \\ \min(x,y) & \text{otherwise.} \end{cases}$$

$$U_{\max(\min,\max,e)}(x,y) = \begin{cases} \min(x,y), & \text{if } (x,y) \in [0,e]^2, \\ \max(x,y) & \text{otherwise.} \end{cases}$$

Note that the above examples are idempotent, and are, as a consequence, averaging functions.

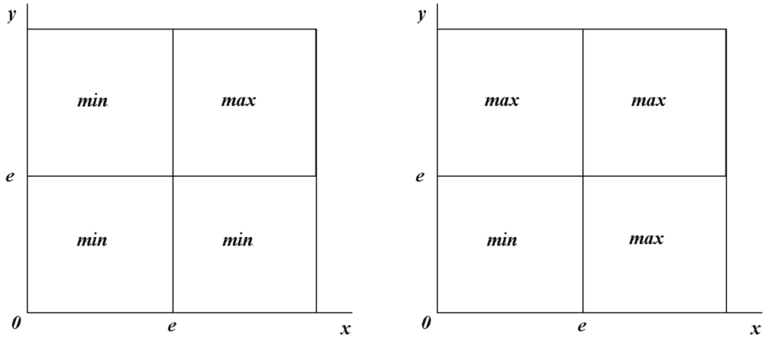


Fig. 4.6. Structure of the idempotent uninorms (conjunctive on the left, disjunctive on the right) in the families $\mathcal{U}_{\min}$ and $\mathcal{U}_{\max}$ (example 4.17, p. 207).

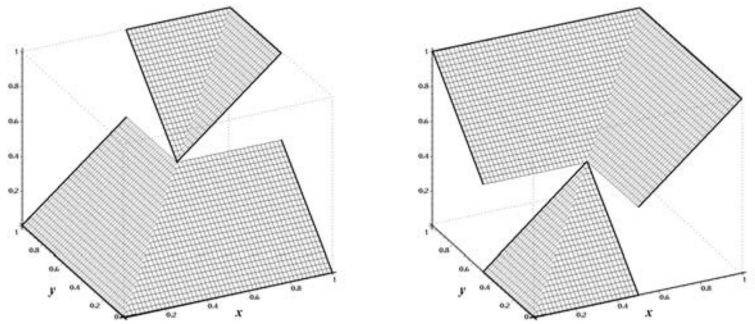


Fig. 4.7. 3D plot of the idempotent uninorms in the families $\mathcal{U}_{\min}$ and $\mathcal{U}_{\max}$ with the neutral element $e = 0.5$ (example 4.17, p. 207).

Example 4.18. An important family of parameterized representable uninorms is given by [99, 141], see also [78],

$$U_\lambda(x,y) = \frac{\lambda xy}{\lambda xy + (1-x)(1-y)} \qquad (x,y) \in [0,1]^2 \backslash \{(0,1),(1,0)\},$$

where $\lambda \in]0,+\infty[$ and either $U_\lambda(0,1) = U_\lambda(1,0) = 0$ (in which case U_λ is conjunctive) or $U_\lambda(0,1) = U_\lambda(1,0) = 1$ (and then U_λ is disjunctive).

U_λ has neutral element $e_\lambda = \frac{1}{1+\lambda}$ and it can be obtained by means of the additive generator

$$u_\lambda(t) = \log\left(\frac{\lambda t}{1-t}\right)$$

The corresponding underlying t–norm and t–conorm are:

$$T_{U_\lambda}(x, y) = \frac{\lambda xy}{\lambda + 1 - (x + y - xy)}$$

and

$$S_{U_\lambda}(x, y) = \frac{x + y + (\lambda - 1)xy}{1 + \lambda xy}$$

which belong to the Hamacher family (see Chapter 3): indeed, a simple calculation shows that T_{U_λ} is the Hamacher t-norm $T^H_{\frac{\lambda+1}{\lambda}}$, whereas S_{U_λ} is the Hamacher t-conorm $S^H_{\lambda+1}$.

Example 4.19 (The $3 - \Pi$ *function).*

Taking $\lambda = 1$ in the above example provides a well-known representable uninorm, usually referred to as the $3 - \Pi$ *function*, which, in the general n-ary case is written as:

$$U(x_1, x_2, \ldots, x_n) = \frac{\prod_{i=1}^{n} x_i}{\prod_{i=1}^{n} x_i + \prod_{i=1}^{n}(1 - x_i)},$$

with the convention $\frac{0}{0} = 0$ if one wants to obtain a conjunctive uninorm, and choosing $\frac{0}{0} = 1$ in order to obtain a disjunctive one. Its additive generator is

$$u(t) = \log\left(\frac{t}{1-t}\right).$$

The $3 - \Pi$ function is a special case of Dombi's aggregative operator [78].

Note that PROSPECTOR's aggregation function in Example 4.5 is precisely the $3 - \Pi$ function when using the $[0, 1]$ scale.

Example 4.20 (MYCIN's aggregation function). MYCIN's aggregation function on $[0, 1]$, (cf. Example 4.4) is a conjunctive representable uninorm with an additive generator [65, 246]

$$u(t) = \begin{cases} \log(2t), & \text{if } t \le \frac{1}{2}, \\ -\log(2 - 2t) & \text{otherwise.} \end{cases} \tag{4.5}$$

It has neutral element $e = \frac{1}{2}$, and the inverse of u is given as

$$u^{-1}(t) = \begin{cases} \frac{e^t}{2}, & \text{if } t \le 0, \\ 1 - \frac{e^{-t}}{2} & \text{otherwise.} \end{cases} \tag{4.6}$$

Thus evaluation of MYCIN's function for any number of inputs can be done by using

$$U(\mathbf{x}) = u^{-1}\left(\sum_{i=1}^{n} u(x_i)\right),$$

with u and u^{-1} given in (4.5) and (4.6).

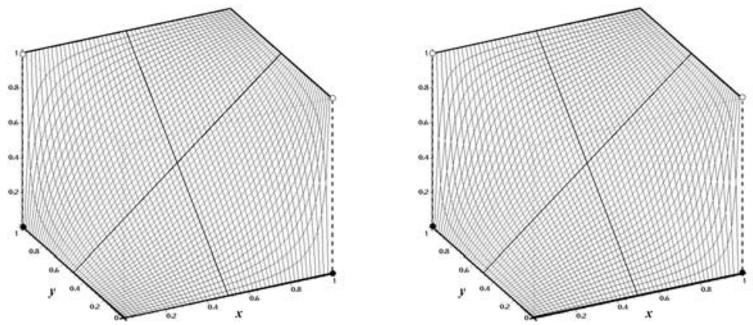


Fig. 4.8. 3D plots of the $3 - \Pi$ (left) and MYCIN's (right) uninorms. The two intersecting straight lines illustrate the presence of the neutral element.

Example 4.21. An interesting example arises in the case of $u(t) = -\log(\log_\lambda(t))$, where $\lambda \in (0, 1)$. u is a strictly increasing bijection, and $u(\lambda) = 0$. The inverse is $u^{-1}(t) = \lambda^{\exp(-t)}$. Then

$$U(\mathbf{x}) = \lambda^{\prod_{i=1}^{n} \log_\lambda x_i} = \exp\left((\log \lambda)^{1-n} \prod_{i=1}^{n} \log x_i\right)$$

is a representable uninorm with the neutral element λ.[9] In the special case $n = 2$ we obtain an alternative compact expression

$$U(x, y) = x^{\log_\lambda y} = y^{\log_\lambda x},$$

and, of course, we can have either conjunctive ($U(0, 1) = U(1, 0) = 0$) or disjunctive ($U(0, 1) = U(1, 0) = 1$) cases (compare to the mean in the Example 2.26). The limiting cases of $\lambda = 1$ and $\lambda = 0$ correspond to drastic t–norm and t–conorm respectively.

The corresponding underlying (strict) t–norm and t–conorm are given, respectively, by the additive generators $g(t) = -u(\lambda t) = \log(\log_\lambda(\lambda t))$ and $h(t) = u(\lambda + (1-\lambda)t) = -\log(\log_\lambda(\lambda + (1-\lambda)t))$. Interestingly, the underlying t–norm is related to U by the equation

$$T_U(x, y) = xyx^{\log_\lambda y} \;=\; xyU(x, y).$$

The t–norm T_U is a Gumbel-Barnett copula (see Example 3.119, part 4) for $\lambda \in [0, \frac{1}{e}]$, with parameter $a = -\frac{1}{\log \lambda}$.

9 To avoid confusion with the Euler number e, we denote the neutral element by λ in this example.

4.2.5 Calculation

Calculation of numerical values of uninorms can be performed recursively, or, for representable uninorms, by using their additive generators. Similarly to the case of t–norms and t–conorms, care should be taken with respect to evaluation of functions with asymptotic behavior, as well as numerical underflow and overflow.

However, evaluation of uninorms presents an additional challenge: these functions are discontinuous, at the very least at the points $(0, 1)$ and $(1, 0)$. In the surrounding regions, these outputs exhibit instability with respect to input inaccuracies. For example, for a conjunctive uninorm (like MYCIN's or PROSPECTORS's functions), the value $f(0, 1) = 0$, but $f(\varepsilon, 1) = 1$ for any $\varepsilon > 0$, like 0.00001.

Evaluation using additive generators also has its specifics. For t–norms and t–conorms, an additive generator could be multiplied by any positive constant without changing the value of the t–norm. This is also true for the generators of the representable uninorms. However, when an additive generator is defined piecewise, like in Example 4.20, one may be tempted to multiply just one part of the expression aiming at numerical stability. While this does not affect the values of the underlying t–norm or t–conorm in the regions $[0, e]^n$ and $[e, 1]^n$, it certainly affects the values in the rest of the domain (see Note 4.15).

4.2.6 Fitting to the data

We examine the problem of choosing the most suitable uninorm based on empirical data, following the general approach outlined in Section 1.6. We have a set of empirical data, pairs $(\mathbf{x}_k, y_k), k = 1, \ldots, K$, which we want to fit as best as possible by using a uninorm. Our goal is to determine the best function from that class that minimizes the norm of the differences between the predicted $(f(\mathbf{x}_k))$ and observed (y_k) values. We will use the least squares or least absolute deviation criterion, as discussed on p. 33.

For uninorms, this problem has two aspects: fitting the actual uninorm and choosing the value of the neutral element. First we consider the first part, assuming e is fixed or given. Then we discuss fitting the value of e.

If the uninorm is given algebraically, e.g., through parametric families of the underlying t–norms and t–conorms, then we have a generic nonlinear optimization problem of fitting the parameter(s) to the data. In doing so, one has to be aware that even if the underlying t–norm and t–conorm are specified, there is a degree of freedom for the values outside $[0, e]^n$ and $[e, 1]^n$. Therefore a rule for how the values in this part of the domain are determined has to be specified.

For representable uninorms, there is an alternative method based on the additive generators. It can be applied in parametric and non-parametric form. In parametric form, when the algebraic form of the additive generators of the underlying t–norm and t–conorm are specified, we aim at fitting three

parameters, λ_T, λ_S and α, the first two are the parameters identifying a specific member of the families of t–norms/t–conorms, and α is the parameter which determines the values outside $[0,e]^n$ and $[e,1]^n$.

Specifically, let the underlying t–norm and t–conorm have additive generators g_{λ_T} and h_{λ_S}. The additive generator of the uninorm with the neutral element e has the form

$$u(t;\lambda_T,\lambda_S,\alpha) = \begin{cases} -g_{\lambda_T}(t/e), & \text{if } t \le e, \\ \alpha h_{\lambda_S}(\frac{t-e}{1-e}) & \text{otherwise.} \end{cases}$$

Note that for any $\alpha > 0$ u generates a uninorm with exactly the same underlying t–norm and t–conorm, but the values outside $[0,e]^n$ and $[e,1]^n$ depend on α. Therefore, the least squares or LAD criterion has to be minimized with respect to variables λ_T, λ_S and $\alpha > 0$. This is a nonlinear optimization problem with possibly multiple locally optimal solutions. We recommend using global optimization methods discussed in the Appendix A.5.4 and A.5.5.

The nonparametric approach is similar to the one used to fit additive generators of t–norms in Section 3.4.15. We represent the additive generator with a monotone increasing regression spline

$$S(t) = \sum_{j=1}^{J} c_j B_j(t), \tag{4.7}$$

with the basis functions B_j chosen in such a way that the monotonicity condition is expressed as a set of linear inequalities (see [13, 15]). Our goal is to determine from the data the unknown coefficients $c_j > 0$ (for monotone increasing S), subject to conditions

$$S(e) = \sum_{j=1}^{J} c_j B_j(e) = 0, \quad S(a) = \sum_{j=1}^{J} c_j B_j(a) = 1.$$

Since the underlying t–norm and t–conorms are necessarily strict, we use the well-founded generators [136, 137], defined as

$$u(t) = \begin{cases} -(\frac{1}{t} + S(\varepsilon_1) - \frac{1}{\varepsilon_1}), & \text{if } t \le \varepsilon_1, \\ \frac{1}{1-t} + S(1-\varepsilon_2) - \frac{1}{\varepsilon_2}, & \text{if } 1-t \le \varepsilon_2, \\ S(t) & \text{otherwise.} \end{cases} \tag{4.8}$$

The values of $\varepsilon_1, \varepsilon_2$ are chosen in such a way that ε_1 is smaller than the smallest strictly positive number out of $x_{ik}, y_k, i = 1, \dots, n_k, k = 1 \dots, K$, and ε_2 is smaller than the smallest strictly positive number out of $1 - x_{ik}, 1 - y_k, i = 1, \dots, n_k, k = 1 \dots, K$. The value a can be chosen as $a = \min\{\varepsilon_1, 1 - \varepsilon_2\}$.

Fitting the coefficients is performed by solving a quadratic programming problem (in the case of LS criterion)

$$\begin{aligned}
\text{Minimize} \quad & \sum_{k=1}^{K} \left(\sum_{j=1}^{J} c_j B_j(\mathbf{x}_k, y_k) \right)^2 \\
\text{s.t.} \quad & \sum_{j=1}^{J} c_j B_j(e) = 0, \\
& \sum_{j=1}^{J} c_j B_j(a) = 1, \\
& c_j > 0.
\end{aligned} \tag{4.9}$$

or, in the case of LAD criterion,

$$\begin{aligned}
\text{Minimize} \quad & \sum_{k=1}^{K} \left| \sum_{j=1}^{J} c_j B_j(\mathbf{x}_k, y_k) \right| \\
\text{s.t.} \quad & \sum_{j=1}^{J} c_j B_j(e) = 0, \\
& \sum_{j=1}^{J} c_j B_j(a) = 1, \\
& c_j > 0.
\end{aligned} \tag{4.10}$$

The functions B_j are defined as

$$B_j(\mathbf{x}_k, y_k) = B_j(x_{1k}) + B_j(x_{2k}) + \ldots + B_j(x_{n_k k}) - B_j(y_k). \tag{4.11}$$

Standard QP and LP methods are then applied to the mentioned problems.

Next, consider the case when the value of the neutral element e is not given, but has to be computed from the data. In this case one has to perform numerical optimization with respect to e and spline coefficients c_j (or parameters λ_T, λ_S and $\alpha > 0$). This is a complicated constrained global optimization problem. However it can be set as a bi-level optimization problem

$$\min_{e \in [0,1]} \left[\min_{c_j} \sum_{k=1}^{K} \left(\sum_{j=1}^{J} c_j B_j(\mathbf{x}_k, y_k) \right)^2 \text{ s.t. linear conditions on } c_j \right]. \tag{4.12}$$

The inner problem is exactly the same as (4.9) or (4.10), with a fixed e, and the outer problem is a univariate global optimization problem. We recommend using the Pijavki-Shubert deterministic method (see Appendix A.5.5), which requires only the values of the objective function (expression in the brackets). The latter is given as a solution to a QP or LP, which is not expensive numerically. In any case, for reasonable accuracy, the outer problem is not expected to require too many objective function evaluations.

As an example, Fig. 4.9 shows a graph of the additive generator found by solving problem (4.12) with the data generated using the $3 - \Pi$ uninorm (Example 4.19), whose additive generator is also presented for comparison. 60 random data were generated. Note the accuracy with which the value of the neutral element e has been computed.

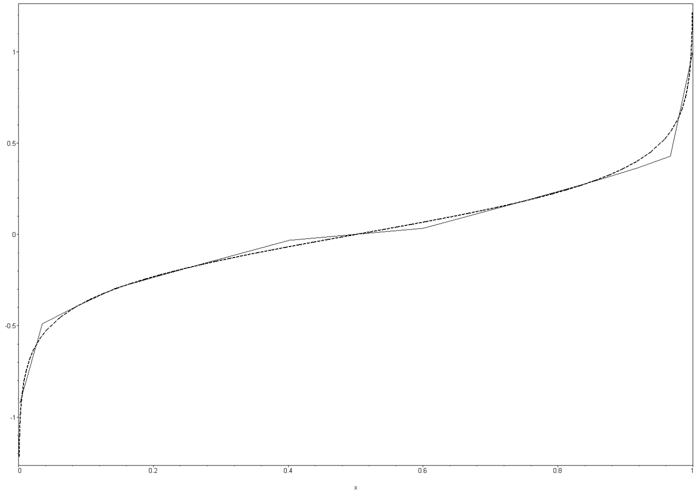


Fig. 4.9. Fitting an additive generator of the $3-\Pi$ uninorm using empirical data. The smooth curve is the true additive generator and the piecewise linear curve is its spline approximation. The computed value of e was 0.498.

4.3 Nullnorms

Uninorms (Section 4.2) are associative and symmetric aggregation functions that act as t-norms when receiving low inputs and as t-conorms when dealing with high values. Nullnorms[10] are associative and symmetric aggregation functions with the opposite behavior, that is, acting as t-conorms for low values and as t-norms for high values. Similarly to uninorms, nullnorms are averaging functions when dealing with mixed inputs (those including both low and high values), but their behavior in such cases is much more restrictive, since it is limited to a unique value (which coincides with the absorbing element).

4.3.1 Definition

In a similar way to uninorms, nullnorms are defined as a generalization of t-norms and t-conorms by just modifying the axiom concerning the neutral element. The definition for the bivariate case is the following:

Definition 4.22 (Nullnorm). *A nullnorm is a bivariate aggregation function* $V : [0,1]^2 \to [0,1]$ *which is associative, symmetric, such that there exists an element* a *belonging to the open interval* $]0,1[$ *verifying*

$$\forall t \in [0,a], \qquad V(t,0) = t, \tag{4.13}$$

$$\forall t \in [a,1], \qquad V(t,1) = t. \tag{4.14}$$

[10] Nullnorms were introduced under this name in 2001 [42], but are equivalent to the so-called t-operators [171].

Note 4.23. The original definition included an additional condition that the element a is the absorbing element of V. This condition is redundant since it follows directly from monotonicity and the conditions (4.13) and (4.14)[11].

Note 4.24. Nullnorms can also be defined allowing the element a to range over the whole interval $[0, 1]$, in which case t-norms and t-conorms become special limiting cases of nullnorms:

- Taking $a = 1$, condition (4.13) states that 0 is the neutral element of V, and then, by definition, V would be a t-conorm.
- Analogously, choosing $a = 0$, condition (4.14) states that 1 is the neutral element of V, and this would entail that V is a t-norm.

Note 4.25. Since nullnorms are associative, they are extended in a unique way to functions with any number of arguments. Thus nullnorms constitute a class of extended aggregation functions as defined in Definition 1.6 on p. 4.

4.3.2 Main properties

Figure 4.1 shows that a nullnorm with absorbing element a is disjunctive in the square $[0, a]^2$ and conjunctive in $[a, 1]^2$. In fact, any nullnorm V with absorbing element a is related to a t-conorm S_V and a t-norm T_V, known as the underlying functions of V, such that:

$$\forall (x, y) \in [0, a]^2, \qquad V(x, y) = a \cdot S_V\left(\frac{x}{a}, \frac{y}{a}\right),$$
$$\forall (x, y) \in [a, 1]^2, \qquad V(x, y) = a + (1 - a) \cdot T_V\left(\frac{x - a}{1 - a}, \frac{y - a}{1 - a}\right).$$

On the remaining parts of the unit square, nullnorms return as the output the absorbing element, i.e:

$$\forall (x, y) \in [0, a] \times [a, 1] \cup [a, 1] \times [0, a], \qquad V(x, y) = a.$$

In particular, it is $V(0, 1) = V(1, 0) = a$. The structure of nullnorms is depicted in Figure 4.10, using the following notation:

$$S_V^*(x, y) = a \cdot S_V\left(\frac{x}{a}, \frac{y}{a}\right),$$
$$T_V^*(x, y) = a + (1 - a) \cdot T_V\left(\frac{x - a}{1 - a}, \frac{y - a}{1 - a}\right).$$

Therefore, similarly to uninorms, each nullnorm univocally defines a t-norm and a t-conorm. The converse (which is false in the case of uninorms), is true for nullnorms, that is, given an arbitrary t-norm T, an arbitrary t-conorm

[11] Indeed, for any $t \in [0, 1]$, it is $a = V(a, 0) \leq V(a, t) \leq V(a, 1) = a$, and this implies $V(a, t) = a$.

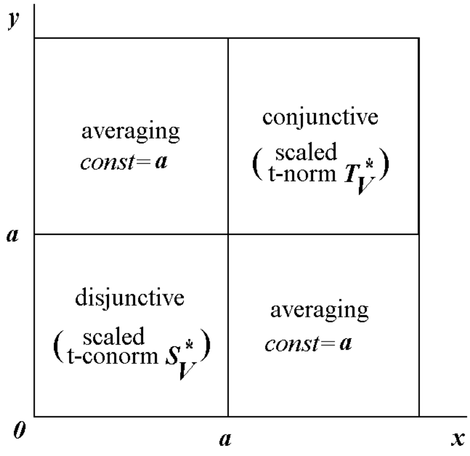


Fig. 4.10. Structure of a nullnorm with absorbing element a, underlying t-norm T_V and underlying t-conorm S_V.

S and an element $a \in]0,1[$, there is a unique nullnorm V with absorbing element a such that $T_V = T$ and $S_V = S$. Such a nullnorm is given by

$$V_{T,S,a}(x,y) = \begin{cases} a \cdot S\left(\frac{x}{a}, \frac{y}{a}\right), & \text{if } (x,y) \in [0,a]^2, \\ a + (1-a) \cdot T\left(\frac{x-a}{1-a}, \frac{y-a}{1-a}\right), & \text{if } (x,y) \in [a,1]^2, \\ a & \text{otherwise.} \end{cases}$$

Thanks to associativity, the above result can be readily extended to the general n-ary case as follows:

$$V_{T,S,a}(x_1,\ldots,x_n) = \begin{cases} a \cdot S\left(\frac{x_1}{a}, \ldots, \frac{x_n}{a}\right), & \text{if } (x_1,\ldots,x_n) \in [0,a]^n, \\ a + (1-a) \cdot T\left(\frac{x_1-a}{1-a}, \ldots, \frac{x_n-a}{1-a}\right), & \text{if } (x_1,\ldots,x_n) \in [a,1]^n, \\ a & \text{otherwise.} \end{cases}$$

Observe that *any* couple (t-norm, t-conorm) may be used to construct a nullnorm, independently of the properties that the t-norm and the t-conorm exhibit: for example, it is possible to choose two nilpotent functions, or two strict ones, or a mixed couple made of one nilpotent and one strict aggregation function, and so on. Of course, since both t-norms and t-conorms can be parameterized, the same happens to nullnorms.

The main properties of nullnorms are easily deduced from their structure:

Absorbing element As it was already mentioned in Note 4.23, each nullnorm has the absorbing element a.

Duality The class of nullnorms is closed under duality, that is, the dual function of a nullnorm V with respect to an arbitrary strong negation N, defined as

$$V_d(x_1,\dots,x_n) = N(V(N(x_1),\dots,N(x_n))),$$

is also a nullnorm, and if V has absorbing element a, V_d has absorbing element $N(a)$[12]. Contrary to the case of uninorms, it is possible to find self-dual nullnorms, but a clear necessary condition for this is that a verifies $a = N(a)$[13]. More specifically [171], a nullnorm $V_{T,S,a}$ is self-dual w.r.t. a strong negation N if and only if the following two conditions hold:

1. $N(a) = a$;
2. $\forall (x_1,\dots,x_n) \in [0,1]^n : \ S(x_1,\dots,x_n) = \tilde{N}^{-1}\Big(T(\tilde{N}(x_1),\dots,\tilde{N}(x_n))\Big)$, where $\tilde{N} : [0,1] \to [0,1]$ is the strict negation defined by $\tilde{N}(t) = \frac{N(t\cdot a)-a}{1-a}$ (i.e., the underlying t-norm and t-conorm are dual w.r.t. the strict negation $\tilde{N}$).

Continuity Nullnorms are continuous if and only if the underlying t-norm and t-conorm are continuous.

Idempotency The only idempotent nullnorms are those related to the unique idempotent t-norm, min, and the unique idempotent t-conorm, max, that is, given a value $a \in]0,1[$, there is only one idempotent nullnorm with absorbing element a:

$$V_{\min,\max,a}(x_1,\dots,x_n) = \begin{cases} \max(x_1,\dots,x_n), & \text{if } (x_1,\dots,x_n) \in [0,a]^n, \\ \min(x_1,\dots,x_n), & \text{if } (x_1,\dots,x_n) \in [a,1]^n, \\ a & \text{otherwise.} \end{cases}$$

Idempotent nullnorms are nothing but the extended averaging functions known as a-medians, already discussed in Section 2.8.1. Note that these functions are self-dual with respect to any strong negation N with fixed point a. A plot of one such function is given in Fig. 4.13.

Strict monotonicity Nullnorms may only be strictly monotone in the open squares $]0,a[^n$ and $]a,1[^n$, and this will happen only when the underlying t-norm and t-conorm are strictly monotone.

4.3.3 Examples

In the following we describe some prototypical examples of nullnorms:

Example 4.26 (Weakest nullnorm). Given $a \in]0,1[$, the weakest nullnorm with absorbing element a is the one constructed by means of the weakest t-norm, T_D, and the weakest t-conorm, max. It has the following structure:

$$V_{T_D,\max,a}(x_1,\dots,x_n) = \begin{cases} \max(x_1,\dots,x_n), & \text{if } (x_1,\dots,x_n) \in [0,a]^n, \\ x_i, & \text{if } x_i \ge a \text{ and } x_j = 1 \text{ for all } j \ne i, \\ a & \text{otherwise.} \end{cases}$$

[12] For details on the structure of V_d see, e.g., [171].
[13] In other words, a must coincide with the fixed point of the negation N.

Example 4.27 (Strongest nullnorm). Given $a \in]0,1[$, the strongest nullnorm with absorbing element a is the one constructed by means of the strongest t-norm, min, and the strongest t-conorm, S_D:

$$V_{\min,S_D,a}(x_1,\ldots,x_n) = \begin{cases} \min(x_1,\ldots,x_n), & \text{if } (x_1,\ldots,x_n) \in [a,1]^n, \\ x_i, & \text{if } x_i \leq a \text{ and } x_j = 0 \text{ for all } j \neq i, \\ a & \text{otherwise.} \end{cases}$$

The structure of these two limiting nullnorms can be viewed, for the bivariate case, in Figure 4.11. See also Figure 4.12 for a visualization of the corresponding 3D plots when choosing $a = 0.5$ as absorbing element.

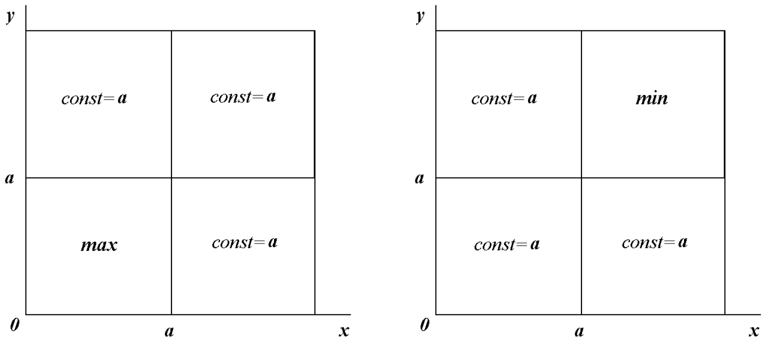


Fig. 4.11. Structure of the weakest (left) and the strongest (right) bivariate nullnorms (examples 4.26 and 4.27) with absorbing element a.

Example 4.28 (Łukasiewicz nullnorm). If the underlying t-norm and t-conorm of a nullnorm are taken, respectively, as the Łukasiewicz t–norm and t–conorm T_L and S_L (see Chapter 3), the following function is obtained in the bivariate case (see Figure 4.13):

$$V_{T_L,S_L,a}(x,y) = \begin{cases} x+y, & \text{if } x+y \leq a, \\ x+y-1, & \text{if } x+y \geq 1+a, \\ a & \text{otherwise.} \end{cases}$$

4.3.4 Calculation

Calculation of the values of nullnorms is performed based on the scaled versions of the underlying t–norm and t–conorm, with the constant value a assigned outside the region $[0,a]^n \cup [a,1]^n$. Calculation can be performed using recursive formulas, or by using additive generators. We refer to Section 3.4.13.

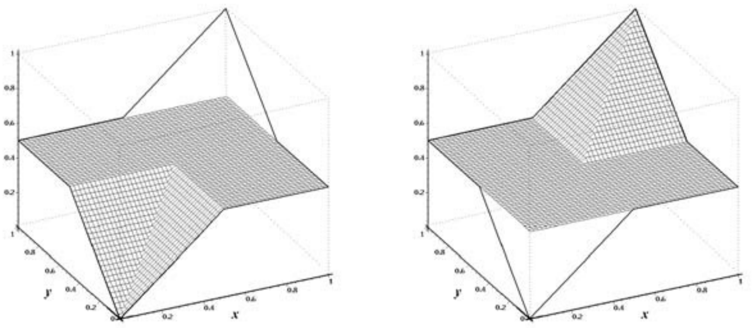


Fig. 4.12. 3D plots of the weakest and the strongest bivariate nullnorms (examples 4.26 and 4.27) with absorbing element $a = 0.5$

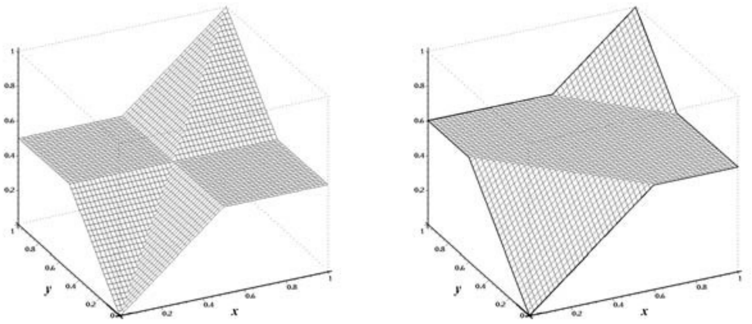


Fig. 4.13. 3D plots of the bivariate idempotent nullnorm $V_{\min,\max,a}$ with absorbing element $a = 0.5$ (left) and Łukasiewicz nullnorm (example 4.28) with absorbing element $a = 0.6$ (right).

4.3.5 Fitting to the data

We follow the same approach taken in Section 4.2.6, p. 211, to fit uninorms to empirical data. We can use either parametric or non-parametric methods. However, an important distinction from uninorms is that nullnorms are defined uniquely by the underlying t–norm and t–conorm and the value of the annihilator a. There is no freedom of choosing values outside the region $[0, a]^n \cup [a, 1]^n$.

In other words, given a fixed value of a, the underlying t–norm and t–conorms are completely independent, and can be fitted independently using

the data which falls into $[0, a]^n$ and $[a, 1]^n$. The underlying t–norm and t–conorms can be either strict or nilpotent, or ordinal sums.

Since any continuous t–norm and t–conorm can be approximated arbitrarily well by an Archimedean one, and continuous Archimedean t–norms and t–conorms are defined by their additive generators, then it makes sense to fit additive generators in the same way as was done for t–norms in Section 3.4.15. To do this, the data is split into three groups: the data on $[0, a]^n$, the data on $[a, 1]^n$ and the data elsewhere (group 3). The first two groups are used to fit (scaled) additive generators of the underlying t–norm and t–conorm. The third group is discarded.

Next consider fitting the unknown value of the annihilator a to the data. Here we vary a over $[0, 1]$ to minimize the least squares or LAD criterion. Now all data (including group 3) are utilized in calculating the differences between the predicted and observed values. This can be set as a bi-level optimization problem, where at the outer level we minimize

$$\min_{a \in [0,1]} \sum_{k=1,\dots,K} \left(f_a(\mathbf{x}_k) - y_k\right)^2,$$

where $f_a(\mathbf{x})$ is the nullnorm computed from the underlying scaled t–norm and t–conorm and a fixed parameter a, i.e., f_a is the solution to problem of type (3.18) or (3.19) with an appropriate scaling.

4.4 Generated functions

We have studied in Chapters 2–4 several types of aggregation procedures which follow essentially the same pattern:

- transform each input value using (possibly different) univariate functions;
- add the transformed values;
- return as the output some transformation of that sum.

The term *transformation* is understood as application of a univariate function, fulfilling some basic properties. Such aggregation functions are known as *generated functions*, or *generated operators*. The role of generators is played by the univariate functions used to transform the input and output values.

Several generated functions have already been studied in this book, namely weighted quasi-arithmetic means, Archimedean triangular norms and conorms and representable uninorms (see Section 4.4.1 below). This means that the class of generated functions comprises extended aggregation functions with all kinds of behavior, that is, there are averaging, conjunctive, disjunctive and mixed generated functions. In this section we summarize the most general properties of generated functions, and study three specific classes of mixed aggregation functions, not mentioned earlier.

4.4.1 Definitions

Definition 4.29 (Generated function). *Let $g_1, \ldots, g_n : [0,1] \to [-\infty, +\infty]$ be a family of continuous non-decreasing functions and let $h : \sum_{i=1}^n Ran(g_i) \to [0,1]$ be a continuous non-decreasing surjection*[14]*. The function $f : [0,1]^n \to [0,1]$ given by*

$$f(x_1, \ldots, x_n) = h\left(g_1(x_1) + \ldots + g_n(x_n)\right)$$

is called a generated function, and $(\{g_i\}_{i \in \{1,\ldots,n\}}, h)$ is called a generating system.

Note 4.30. Observe that if there exist $j, k \in \{1, \ldots, n\}$, $j \neq k$, such that $g_j(0) = -\infty$ and $g_k(1) = +\infty$, then the generated function is not defined at the points such that $x_j = 0$ and $x_k = 1$, since the summation $-\infty + \infty$ or $+\infty - \infty$ appears in these cases. When this occurs, a convention, such as $-\infty + \infty = +\infty - \infty = -\infty$, is adopted, and – see Section 4.4.2 below – continuity of f is lost.

The monotonicity of the functions that form the generating system, along with the fact that h is surjective, provides the following result:

Proposition 4.31. *Every generated function is an aggregation function in the sense of Definition 1.5.*

Several important families of generated functions have already been analyzed in this book. In particular, recall the following families:

- *Continuous Archimedean t-norms*[15] (Proposition 3.37)

$$T(\mathbf{x}) = g^{(-1)}(g(x_1) + \ldots + g(x_n)),$$

  where $g : [0,1] \to [0, \infty]$, $g(1) = 0$, is a continuous strictly decreasing function and $g^{(-1)}$ is its pseudo-inverse, are generated functions with generating system given by $g_i(t) = -g(t)$ and $h(t) = -g^{(-1)}(t)$.
- *Continuous Archimedean t-conorms* (Proposition 3.45)

$$S(\mathbf{x}) = g^{(-1)}(g(x_1) + \ldots + g(x_n)),$$

  where $g : [0,1] \to [0, \infty]$, $g(0) = 0$, is a continuous strictly increasing function and $g^{(-1)}$ is its pseudo-inverse, are generated functions with generating system given by $g_i(t) = g(t)$ and $h(t) = g^{(-1)}(t)$.

---

[14] That is, $Ran(h) = [0,1]$.

[15] In particular, Archimedean copulas.

- *Weighted Archimedean t-norms* (Definition 3.91)

$$T_{\mathbf{w}}(\mathbf{x}) = g^{(-1)}(w_1 g(x_1) + \ldots + w_n g(x_n)),$$

  where $g : [0,1] \to [0,\infty]$, $g(1) = 0$, is a continuous strictly decreasing function, $g^{(-1)}$ is its pseudo-inverse, and $\mathbf{w}$ is a weighting vector, are generated functions with generating system given by $g_i(t) = -w_i g(t)$ and $h(t) = -g^{(-1)}(t)$.
- *Weighted Archimedean t-conorms* (Definition 3.98)

$$S_{\mathbf{w}}(\mathbf{x}) = g^{(-1)}(w_1 g(x_1) + \ldots + w_n g(x_n)),$$

  where $g : [0,1] \to [0,\infty]$, $g(0) = 0$, is a continuous strictly increasing function, $g^{(-1)}$ is its pseudo-inverse, and $\mathbf{w}$ is a weighting vector, are generated functions with generating system given by $g_i(t) = w_i g(t)$ and $h(t) = g^{(-1)}(t)$.
- *Weighted quasi-arithmetic means* (Definition 2.18)

$$M_{\mathbf{w},g}(\mathbf{x}) = g^{-1}(w_1 g(x_1) + \ldots + w_n g(x_n)),$$

  where $g : [0,1] \to [-\infty,+\infty]$ is a continuous strictly monotone function and $\mathbf{w}$ is a weighting vector, are generated functions with generating system given by $g_i(t) = w_i g(t)$ and $h(t) = g^{-1}(t)$ (or $g_i(t) = -w_i g(t)$ and $h(t) = -g^{-1}(t)$ if g is decreasing).
- *Representable uninorms* (Definition 4.11)

$$U(\mathbf{x}) = u^{-1}\left(u(x_1) + \ldots + u(x_n)\right), \quad \{0,1\} \nsubseteq \{x_1, \ldots, x_n\},$$

  where $u : [0,1] \to [-\infty,+\infty]$ is a strictly increasing bijection, are generated functions with generating system given by $g_i(t) = u(t)$ and $h(t) = u^{-1}(t)$.

Note 4.32. Observe that different generating systems may define the same generated function. A simple example of this situation is given by the generating systems $(g_i(t), h(t))$ and $(cg_i(t), h(\frac{t}{c}))$, $c \in \Re, c > 0$, that generate the same function. Another example is provided by weighted quasi-arithmetic means, which, as mentioned in Chapter 2, are equivalent up to affine transformations of their generating functions g, and this translates into different generating systems.

Note 4.33. Weighted quasi-arithmetic means are special cases of generated functions. They include weighted arithmetic means, obtained when choosing $g(t) = t$, with associated generating systems of the form $g_i(t) = w_i t$ and $h(t) = t$. Weighted arithmetic means form a special class of generated functions, since a generated function is a weighted arithmetic mean if and only if it can be generated by an affine generating system [151] (i.e., a system such that $g_i(t) = a_i t + b_i$ and $h(t) = at + b$, with $a, a_i, b, b_i \in \Re$, $a, a_i \geq 0$).

Note that Definition 4.29 can be easily upgraded to *families* of generated aggregation functions as follows:

Definition 4.34 (Extended generated aggregation function). *An extended function*

$$F : \bigcup_{n \in \{1,2,\ldots\}} [0,1]^n \to [0,1],$$

verifying $F(t) = t$ *for any* $t \in [0,1]$, *and such that all its restrictions to* $[0,1]^n$, $n > 1$, *are generated functions, is called an extended generated aggregation function.*

4.4.2 Main properties

Continuity Generated functions are always continuous on their whole domain except for some very specific situations involving the aggregation of contradictory information (tuples containing a 0 and a 1). More precisely:
- Generated functions such that $Dom(h) \neq [-\infty, +\infty]$ are always continuous on the whole unit cube.
- Generated functions such that $Dom(h) = [-\infty, +\infty]$ are continuous except for the case when there exist $j, k \in \{1, \ldots, n\}, j \neq k$, such that $g_j(0) = -\infty$ and $g_k(1) = +\infty$, in which case the function generated by the system $(\{g_i\}_{i \in \{1,\ldots,n\}}, h)$ is discontinuous at the points $\mathbf{x}$ such that $x_j = 0$ and $x_k = 1$. An example of this situation is given by representable uninorms (Section 4.2). Note however (see Example 4.45 below) that the condition $Dom(h) = [-\infty, +\infty]$ by itself does not necessarily entail lack of continuity.

Symmetry Generated functions may clearly be either symmetric or asymmetric. Moreover, a generated function is symmetric if and only if it can be generated by a generating system such that for all $i \in \{1, \ldots, n\}$ it is $g_i = g$, where $g : [0,1] \to [-\infty + \infty]$ is any continuous non-decreasing function [151].

Idempotency There are many different classes of non-idempotent generated functions (e.g., continuous Archimedean t-norms/t-conorms or representable uninorms). There are also different idempotent generated functions, such as weighted quasi-arithmetic means. In fact, a generating system $(\{g_i\}_{i \in \{1,\ldots,n\}}, h)$ provides an idempotent generated function if and only if $h^{-1}(t) = \sum_{i=1}^{n} g_i(t)$ for any $t \in [0,1]$ [151].

Note 4.35. This result, along with the result about the symmetry, entails that quasi-arithmetic means are the only idempotent symmetric generated functions.

Duality The class of generated functions is closed under duality with respect to any strong negation. Indeed, given a strong negation N, the dual of a generated function with generating system $(\{g_i\}_{i \in \{1,\ldots,n\}}, h)$ is a function generated by the system $(\{(g_i)_d\}_{i \in \{1,\ldots,n\}}, h_d)$ where

$$(g_i)_d(t) = -g_i(N(t))$$
$$h_d(t) = N(h(-t))$$

Of course, this allows for N-self-dual generated functions. Some of them have already been mentioned in this book, such as weighted quasi-arithmetic means built from bounded generating functions (see Section 2.3) or representable uninorms up to the tuples containing the values 0 and 1 (Proposition 4.14).

Neutral element Some generated functions, such as the ones belonging to the classes of t-norms, t-conorms and uninorms, possess a neutral element, whereas others, such as the arithmetic mean, do not. The following full characterization of the class of extended generated functions possessing a (strong) neutral element is available [151]:

Proposition 4.36. *An extended generated aggregation function F has a neutral element $e \in [0,1]$ if and only if for each $n > 1$ the restriction to $[0,1]^n$ of F, f_n, can be expressed as*

$$f_n(x_1, \ldots, x_n) = g^{(-1)}(g(x_1) + \ldots + g(x_n)) \tag{4.15}$$

where $g : [0,1] \to [-\infty + \infty]$ is a continuous non-decreasing function such that $g(e) = 0$ with the pseudo-inverse (see 3.4.6) $g^{(-1)}$.

The n-ary generated functions of the above characterization can be classified as follows:

1. If $e = 0$, then f_n is a continuous Archimedean t-conorm.
2. If $e = 1$, then f_n is a continuous Archimedean t-norm.
3. If $e \in]0,1[$:
   a) If $Ran(g) = [-\infty + \infty]$, f_n is a representable uninorm, and, hence, associative but non-continuous at the points that simultaneously contain 0 and 1.
   b) Otherwise, f_n is continuous and non-associative (see next section).

Note 4.37. The characterization given in Proposition 4.36 implies that generated functions with a neutral element are necessarily symmetric.

Associativity Instances of associative as well as non-associative generated functions have been given in Section 4.4.1. Among the associative functions, one finds either continuous (e.g., t-norms and t-conorms) or non-continuous (e.g., uninorms) functions. Regarding continuous associative generated functions, it is known that these can only be projection functions, Archimedean t-norms, Archimedean t-conorms or nullnorms [46].

4.4.3 Classes of generated functions

The following classes of generated functions have been already studied in detail:

- Weighted quasi-arithmetic means (Section 2.3);
- Continuous Archimedean t–norms and t–conorms (Section 3.4.4);
- Weighted Archimedean t–norms and t–conorms (Section 3.4.16);
- Representable uninorms (Section 4.2.3).

In this section we examine three other classes of mixed generated aggregation functions. The first two classes – continuous generated functions with a neutral element and weighted uninorms – are related to t-norms and t-conorms.

Continuous generated functions with a neutral element

Recall from Section 4.2, Proposition 4.14, that representable uninorms – class 3.a) above – behave as strict t-norms on $[0, e]^n$ and as strict t-conorms on $[e, 1]^n$. Unfortunately, uninorms are discontinuous, which implies that small changes to the inputs (in the neighborhood of discontinuity) lead to large changes in the output values. Discontinuity is the price for associativity [46, 99], and several authors abandoned associativity requirement (e.g., Yager's generalized uninorm operator GenUNI [270]).

We note that associativity is not required to define aggregation functions for any number of arguments, i.e., extended aggregation functions. Generated aggregation functions are an example of an alternative way. We shall now explore a construction similar to that of uninorms, which delivers continuous mixed aggregation functions, case 3.b) above.

Thus we consider aggregation functions defined by

$$f(\mathbf{x}) = g^{(-1)}(g(x_1) + \ldots + g(x_n)), \tag{4.16}$$

with $g : [0, 1] \to [-\infty + \infty]$ a continuous non-decreasing function such that $g(e) = 0$, $g^{(-1)}$ its pseudo-inverse, and $Ran(g) \subsetneq [-\infty + \infty]$ [180, 184].

According to Proposition 4.36, f has neutral element e. Moreover, it is continuous on $[0, 1]^n$, it acts on $[0, e]^n$ as a continuous scaled t-norm T built from the additive generator $g_T(t) = -g(et)$, and it acts on $[e, 1]^n$ as a continuous scaled t-conorm S built from the additive generator $g_S(t) = g(e+(1-e)t)$.

Note 4.38. Either T, or S, or both are necessarily nilpotent (if both are strict, $Ran(g) = [-\infty + \infty]$ and we obtain a representable uninorm).

Conversely, given a value $e \in]0, 1[$, a continuous t-norm T with an additive generator g_T and a continuous t-conorm S with an additive generator g_S, the mapping $g : [0, 1] \to [-\infty, +\infty]$ given by

$$g(t) = \begin{cases} -g_T\left(\frac{t}{e}\right), & \text{if } t \in [0, e], \\ g_S\left(\frac{t-e}{1-e}\right), & \text{if } t \in]e, 1], \end{cases}$$

defines a generated aggregation function by (4.16), with the neutral element e. If either T or S or both are nilpotent, then f is continuous.

Note 4.39. Since the additive generators of T and S are defined up to an arbitrary positive multiplier, it is possible to use different g_T and g_S which produce the same t-norm and t-conorm on $[0,e]^n$ and $[e,1]^n$, but different values on the rest of the domain. Thus we can use

$$g(t) = \begin{cases} -ag_T\left(\frac{t}{e}\right), & \text{if } t \in [0,e], \\ bg_S\left(\frac{t-e}{1-e}\right), & \text{if } t \in]e,1], \end{cases}$$

with arbitrary $a, b > 0$.

Examples 4.41-4.43 on pp. 227-229 illustrate continuous generated functions with a neutral element.

Weighted generated uninorms

Recall the method of introducing weights in t-norms and t-conorms presented in Section 3.4.16, which is based on using

$$T_{\mathbf{w}}(\mathbf{x}) = g^{(-1)}\left(\sum_{i=1}^{n} w_i g(x_i)\right),$$

with a weighting vector $\mathbf{w}$, not necessarily normalized[16] [47, 49, 261, 272].

Yager [272] and Calvo and Mesiar [48] extended this approach to weighted uninorms, defined as

---

Definition 4.40 (Weighted generated uninorm). *Let* $u : [0,1] \to [-\infty, \infty]$ *be an additive generator of some representable uninorm* U*, and* $\mathbf{w} : w_i \geq 0$ *be a weighting vector (not necessarily normalized). The weighted generated uninorm is defined as*

$$U_{\mathbf{w}}(\mathbf{x}) = u^{-1}\left(\sum_{i=1}^{n} w_i u(x_i)\right). \tag{4.17}$$

Recall from Section 3.4.16 that introduction of weights can be expressed through an importance transformation function H defined by

$$H(w,t) = u^{(-1)}(wu(t)),$$

so that

$$U_{\mathbf{w}} = U(H(w_1, x_1), \ldots, H(w_n, x_n)),$$

provided that $\max w_i = 1$. The function $H : [0,1]^2 \to [0,1]$ in the case of uninorms should satisfy the following properties:

- $H(1,t) = u^{(-1)}(u(t)) = t$;
- $H(0,t) = u^{(-1)}(0) = e$;

---

[16] I.e., a weighting vector such that the condition $\sum w_i = 1$ is not mandatory.

- $H(w,e) = u^{(-1)}(0) = e$;
- $H(w,t)$ is non-decreasing in t and is non-increasing in w when $t \le e$ and non-decreasing if $t \ge e$.

Example 4.44 on p. 229 illustrates a weighted generated uninorm. A weighted uninorm is not associative nor symmetric, and it does not have a neutral element (except in the limiting case $w_i = 1,\ i = 1, \dots, n$).

The approach to introducing weights using additive generators is easily extended to the class of continuous generated functions with a neutral element $e \in]0,1[$ given by (4.16). It is sufficient to take

$$f_{\mathbf{w}}(\mathbf{x}) = g^{(-1)}\left(\sum_{i=1}^{n} w_i g(x_i)\right), \tag{4.18}$$

which differs from (4.17) because it uses pseudoinverse of g. Since $Ran(g) \neq [-\infty, \infty]$, this aggregation function is continuous.

Asymmetric generated functions

Finally, consider the case when the generating functions g_i in Definition 4.29 have different analytical form. Of course, the resulting generated function will be asymmetric. This construction gives rise to a large number of parametric families of aggregation functions of different types. An example of such a function is presented in Example 4.45 on p. 230.

4.4.4 Examples

Continuous generated functions with a neutral element

Example 4.41. [43, 180, 184] Proposition 4.36 allows one to easily build generated functions with a given neutral element. For example, let $e \in [0,1]$ and consider the function $g : [0,1] \to [-\infty, +\infty]$, defined as $g(t) = t - e$, which is continuous, non-decreasing and such that $g(e) = 0$. It is $Ran(g) = [g(0), g(1)] = [-e, 1-e]$, and its pseudo-inverse $g^{(-1)}$ is given by

$$g^{(-1)}(t) = \begin{cases} 1, & \text{if } t > 1-e, \\ t+e, & \text{if } -e \le t \le 1-e, \\ 0, & \text{if } t < -e. \end{cases}$$

Then the function $f : [0,1]^n \to [0,1]$ defined by (4.16) is given by

$$f(x_1, \dots, x_n) = \begin{cases} 1, & \text{if } \sum\limits_{i=1}^{n}(x_i - e) + e > 1, \\ \sum\limits_{i=1}^{n}(x_i - e) + e, & \text{if } 0 \le \sum\limits_{i=1}^{n}(x_i - e) + e \le 1, \\ 0, & \text{if } \sum\limits_{i=1}^{n}(x_i - e) + e < 0, \end{cases}$$

which alternatively can be expressed as

$$f(x_1, \ldots, x_n) = \max\left(0, \min\left(1, \sum_{i=1}^{n}(x_i - e) + e\right)\right).$$

f is a generated aggregation function with neutral element e. If $e = 0$ it is nothing but the Łukasiewicz t-conorm, whereas $e = 1$ provides the Łukasiewicz t-norm. Otherwise, since $Ran(g) \neq [-\infty + \infty]$, f is continuous but is not associative. Further, on $[0, e]^n$ it acts as the continuous t-norm with additive generator $g_T(t) = -g(te) = (1 - t)e$ and on $[e, 1]^n$ it acts as the continuous t-conorm with additive generator $g_S(t) = g(e + (1 - e)t) = (1 - e)t$, i.e., it is an ordinal sum of the Łukasiewicz t-norm and the Łukasiewicz t-conorm [17].

A 3D plot of this function in the bivariate case with $e = 0.5$ is presented on Figure 4.14.

Example 4.42. Let us define a function f in (4.16) using the generating function

$$g(t) = \begin{cases} t - e, & \text{if } 0 \leq t < e, \\ 2(t - e), & \text{if } e \leq t \leq 1. \end{cases}$$

On $[0, e]^n$ and $[e, 1]^n$ f acts, respectively, as the Łukasiewicz t-norm and t-conorm, but it differs from the function in Example 4.41 on the rest of the domain, see Note 4.39.

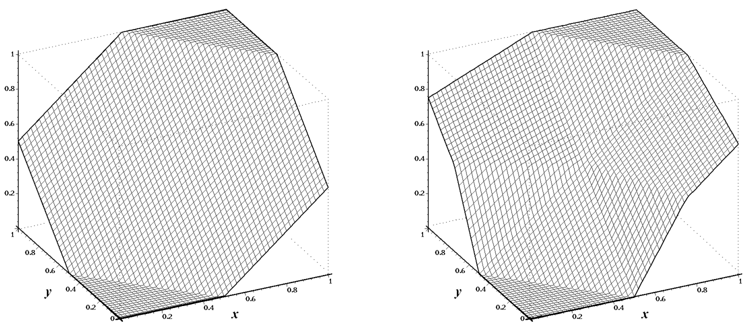


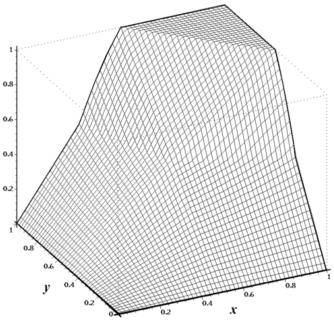
Fig. 4.14. 3D plots of the generated aggregation functions in Examples 4.41 (left) and 4.42 (right) with $e = 0.5$.

[17] Here we use the term "ordinal sum" informally, in the sense that f acts as Łukasiewicz t-norm and t-conorm on $[0, e]^n$ and $[e, 1]^n$ only. In a more formal sense, ordinal sum implies that f is either minimum or maximum on the rest of the domain, see Section 3.4.9.

Example 4.43. Consider a mixed aggregation function which acts as product on $[0,e]^n$ and as Łukasiewicz t-conorm on $[e,1]^n$. One such function is given by (4.16) with

$$g(t) = \begin{cases} \log(t/e), & \text{if } 0 \le t < e, \\ t - e, & \text{if } e \le t \le 1. \end{cases}$$

Its plot is presented on Figure 4.15. Note that it has the neutral element e and the absorbing element $a = 0$, compare with Note 1.32 on p. 13.



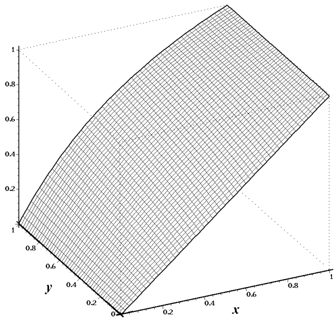


Fig. 4.15. 3D plots of the generated aggregation functions in Example 4.43 with $e = 0.5$ (left) and Example 4.45 (right).

Weighted generated uninorms

Example 4.44. [272] Take the generating function

$$u_\lambda(t) = \log\left(\frac{\lambda t}{1-t}\right),$$

which was used in Example 4.18 on p. 208 to generate representable uninorms

$$U_\lambda(x,y) = \frac{\lambda xy}{\lambda xy + (1-x)(1-y)}, \quad \lambda > 0.$$

The weighted version of this uninorm is given by

$$U_{\lambda,\mathbf{w}}(\mathbf{x}) = \frac{\lambda^{\sum\limits_{i=1}^{n} w_i} \prod\limits_{i=1}^{n} x_i^{w_i}}{\lambda \prod\limits_{i=1}^{n} (1-x_i)^{w_i} + \lambda^{\sum\limits_{i=1}^{n} w_i} \prod\limits_{i=1}^{n} x_i^{w_i}}.$$

Of course, an appropriate convention $\frac{0}{0} = 0$ or $= 1$ is needed.

Asymmetric generated functions

Example 4.45 ([151]). Let $n = 2$, $g_1(t) = \log(\frac{1}{1-t})$, $g_2(t) = \log(1 + t)$ and $h(t) = g_1^{-1}(t) = \frac{e^t}{1+e^t}$. Then the generated function $f(x, y) = h(g_1(x) + g_2(y))$ is given by

$$f(x, y) = \frac{x + xy}{1 + xy}.$$

This function is continuous (although $Dom(h) = [-\infty, +\infty]$), asymmetric and not associative. A 3D plot of this function is presented on Figure 4.15.

4.4.5 Fitting to the data

Fitting continuous generated functions to empirical data can be done using the approach discussed in Section 4.2.6, with the following difference: the asymptotic behavior expressed in (4.8) is no longer needed, and it can be replaced by a condition expressed in (3.18) in the case of nilpotent t–norms, namely $S(a) = -1$.

Fitting parameters of weighted uninorms involves two issues: fitting the weighting vector and fitting an additive generator. When the weighting vector is fixed, the problems of fitting an additive generator is very similar to the one discussed in 4.2.6, except that the functions B_j in (4.11) are multiplied by the corresponding weights. When the generator is fixed, we have the problem analogous to fitting the weights of weighted quasi-arithmetic means, discussed in Section 2.3.7, equation (2.12), except that condition $\sum w_i = 1$ is no longer required. However when both the weighting vector and a generator need to be found, the problem becomes that of global optimization, expressed in (2.19).

4.5 T-S functions

Uninorms and nullnorms (sections 4.2 and 4.3) are built from t-norms and t-conorms in a way similar to ordinal sums, that is, they act as t-norms or as t-conorms in some specific parts of the domain. A rather different approach, still remaining close to t-norms and t-conorms, is the one adopted in the family of functions known as *compensatory T-S functions* (*T-S functions* for short), whose aim is to combine a t-norm and a t-conorm in order to compensate their opposite effects. Contrary to uninorms and nullnorms, T-S functions exhibit a uniform behavior, in the sense that their behavior does not depend on the part of the domain under consideration. They work by separately applying a t-norm and a t-conorm to the given input and then averaging the two values obtained in this way by means of some weighted quasi-arithmetic mean.

The first classes of T-S functions were introduced in [287] with the aim of modeling human decision making processes, and have been studied in more detail and generalized to wider classes of functions [82, 184, 209, 247]. They were first applied in fuzzy linear programming [158] and fuzzy car control [249].

4.5.1 Definitions

The first introduced T-S functions were known as *gamma operators*:

Definition 4.46 (Gamma-operator). *Given $\gamma \in]0,1[$, the gamma aggregation operator with parameter γ is defined as* [18]

$$f_\gamma(x_1,\ldots,x_n) = \left(\prod_{i=1}^{n} x_i\right)^{1-\gamma} \left(1 - \prod_{i=1}^{n}(1-x_i)\right)^{\gamma}$$

Gamma operators perform an exponential combination of two particular functions – the product t-norm T_P and its dual t-conorm S_P, so actually they are special instances of a wider class of functions known as *exponential convex T-S functions*:

Definition 4.47 (Exponential convex T-S function). *Given $\gamma \in]0,1[$, a t-norm T and a t-conorm S, the corresponding exponential convex T-S function is defined as*

$$E_{\gamma,T,S}(x_1,\ldots,x_n) = (T(x_1,\ldots,x_n))^{1-\gamma} \cdot (S(x_1,\ldots,x_n))^{\gamma}$$

Note that the t-norm and the t-conorm involved in the construction of an exponential convex T-S function can be dual to each other, as is the case with gamma operators, but this is not necessary.

Another approach to compensation is to perform a linear convex combination of a t-norm and a t-conorm:

Definition 4.48 (Linear convex T-S function). *Given $\gamma \in]0,1[$, a t-norm T and a t-conorm S, the corresponding linear convex T-S function is defined as*

$$L_{\gamma,T,S}(x_1,\ldots,x_n) = (1-\gamma) \cdot T(x_1,\ldots,x_n) + \gamma \cdot S(x_1,\ldots,x_n).$$

It is clear that both exponential and linear convex T-S functions are obtained from the composition (see Section 1.5) of a t-norm and a t-conorm with a bivariate aggregation function $M_\gamma : [0,1]^2 \to [0,1]$, i.e., they are defined as

$$M_\gamma(T(x_1,\ldots,x_n), S(x_1,\ldots,x_n)),$$

where in the case of exponential functions, $M_\gamma(x,y)$ is given by $x^{1-\gamma}y^\gamma$, whereas for linear functions it is $M_\gamma(x,y) = (1-\gamma)x + \gamma y$. Moreover, the two mentioned outer functions M_γ are particular instances of the family of *weighted quasi-arithmetic means*. In order to see this, let us first recall the definition, in the bivariate case, of these functions (see Definition 2.18 in p. 48 for the general case):

---

[18] This function should not be confused with the gamma function, frequently used in mathematics, which is an extension of the factorial function $n!$ for non-integer and complex arguments. It is defined as $\Gamma(z) = \int_0^\infty t^{z-1}e^{-t}dt$. It has the property $\Gamma(z+1) = z\Gamma(z)$, and since $\Gamma(1) = 1$, we have $\Gamma(n+1) = n!$ for natural numbers n. Its most well known value for non-integer z is $\Gamma(\frac{1}{2}) = \sqrt{\pi}$.

Definition 4.49 (Weighted quasi-arithmetic mean). *Given* $\gamma \in]0,1[$ *and a continuous strictly monotone function* $g : [0,1] \to [-\infty, \infty]$*, the corresponding bivariate weighted quasi-arithmetic mean is defined as*

$$M_{\gamma,g}(x,y) = g^{-1}\big((1-\gamma)g(x) + \gamma g(y)\big)$$

The function g *is called a generating function of* $M_{\gamma,g}$*.*

Clearly the function $M_\gamma(x,y) = x^{1-\gamma}y^\gamma$ is a bivariate weighted quasi-arithmetic mean with generating function $g(t) = \log(t)$ (see Section 2.3), whereas $M_\gamma(x,y) = (1-\gamma)x + \gamma y$ is a bivariate weighted arithmetic mean (with the generating function $g(t) = t$). This observation readily leads to the consideration of a wider class of compensatory functions encompassing both exponential and linear convex T-S functions:

Definition 4.50 (T-S function). *Given* $\gamma \in]0,1[$*, a t-norm* T*, a t-conorm* S *and a continuous strictly monotone function* $g : [0,1] \to [-\infty, \infty]$ *such that* $\{g(0), g(1)\} \neq \{-\infty, +\infty\}$*, the corresponding T-S function is defined as*

$$Q_{\gamma,T,S,g}(x_1,\ldots,x_n) = g^{-1}\Big((1-\gamma)\cdot g(T(x_1,\ldots,x_n)) + \gamma \cdot g(S(x_1,\ldots,x_n))\Big)$$

The function g *is called a generating function of* $Q_{\gamma,T,S,g}$*.*

Note 4.51. An alternative definition allows the parameter γ to range over the whole interval $[0,1]$, and thus includes t–norms (when $\gamma = 0$) and t–conorms (when $\gamma = 1$) as special limiting cases of T-S functions. Observe also that Definition 4.50 excludes, for simplicity, generating functions such that $\{g(0), g(1)\} = \{-\infty, +\infty\}$.

Note 4.52. Due to their construction, and similarly to weighted quasi-arithmetic means (see Section 2.3), different generating functions can lead to the same T-S function: in particular, if $h(t) = ag(t) + b$, where $a, b \in \Re$, $a \neq 0$, a simple calculation shows that $Q_{\gamma,T,S,h} = Q_{\gamma,T,S,g}$.

Exponential convex T-S functions are then nothing but T-S functions with generating function $g(t) = \log(t)$ (or, more generally, $g(t) = a\log(t) + b$ with $a, b \in \Re$, $a \neq 0$, see Note 4.52), whereas linear convex T-S functions are obtained from the class of T-S functions by choosing $g(t) = t$ (or $g(t) = at + b$ with $a, b \in \Re$, $a \neq 0$).

Observe finally that all the above definitions have been given for inputs of any dimension n. This is necessary because the corresponding bivariate functions are not associative. T-S functions may then be used to easily construct extended aggregation functions (as defined in Definition 1.6 on p. 4) sharing the same parameters (γ, T, S, g) for inputs of any dimension $n > 1$:

Definition 4.53 (Extended T-S function). *Given $\gamma \in]0,1[$, a t-norm T, a t-conorm S and a continuous strictly monotone function $g : [0,1] \to [-\infty, \infty]$ such that $\{g(0), g(1)\} \neq \{-\infty, +\infty\}$, the corresponding extended T-S function,*

$$Q_{\gamma,T,S,g} : \bigcup_{n \in \{1,2,\ldots\}} [0,1]^n \to [0,1]$$

is defined as $Q_{\gamma,T,S,g}(t) = t$ for $t \in [0,1]$, and, for any $n > 1$, as

$$Q_{\gamma,T,S,g}(x_1,\ldots,x_n) = g^{-1}\Big((1-\gamma)\cdot g(T(x_1,\ldots,x_n)) + \gamma \cdot g(S(x_1,\ldots,x_n))\Big)$$

4.5.2 Main properties

T-S functions are obviously symmetric and not associative. Other interesting properties of these functions are summarized below:

Bounds Any T-S function is bounded by the t-norm and the t-conorm it is built upon, i.e., the following inequality holds:

$$T \leq Q_{\gamma,T,S,g} \leq S$$

Comparison

- If $\gamma_1 \leq \gamma_2$, then $Q_{\gamma_1,T,S,g} \leq Q_{\gamma_2,T,S,g}$
- If $T_1 \leq T_2$, then $Q_{\gamma,T_1,S,g} \leq Q_{\gamma,T_2,S,g}$
- If $S_1 \leq S_2$, then $Q_{\gamma,T,S_1,g} \leq Q_{\gamma,T,S_2,g}$

Regarding the comparison of T-S functions just differing in their generating functions, it suffices to apply the results on the comparison of weighted quasi-arithmetic means given in section 2.3.

Absorbing element T-S functions have an absorbing element if and only if one of the two following situations hold:

- The generating function g verifies $g(0) = \pm\infty$, in which case the absorbing element is $a = 0$.
- The generating function g verifies $g(1) = \pm\infty$, and then the absorbing element is $a = 1$.

Note that this entails that exponential convex T-S functions – in particular, gamma operators – have absorbing element $a = 0$, whereas linear convex T-S functions do not have an absorbing element.

Neutral element Neither T-S functions nor T-S functions possess a neutral element.

Duality The class of T-S functions is closed under duality, that is, the dual of a function $Q_{\gamma,T,S,g}$ with respect to an arbitrary strong negation N is also a T-S function. Namely, it is given by $Q_{1-\gamma,S_d,T_d,g_d}$, where S_d is the t-norm dual to S w.r.t. N, T_d is the t-conorm dual to T w.r.t. N and $g_d = g \circ N$, that is:

$$S_d(x_1,\dots,x_n) = N(S(N(x_1),\dots,N(x_n)))$$
$$T_d(x_1,\dots,x_n) = N(T(N(x_1),\dots,N(x_n)))$$
$$g_d(t) = g(N(t))$$

Observe that as a corollary we have the following:

- The dual of a T-S function with generating function verifying $g(0) = \pm\infty$ (respectively $g(1) = \pm\infty$) is a T-S function with generating function verifying $g_d(1) = \pm\infty$ (respectively $g_d(0) = \pm\infty$), and thus belonging to a different category. As a consequence, it is clear that these functions (in particular, exponential convex T-S functions) are never self-dual.
- When dealing with functions $Q_{\gamma,T,S,g}$ such that $g(0), g(1) \neq \pm\infty$, the following result regarding self-duality is available ([209]):

  Proposition 4.54. *Let $Q_{\gamma,T,S,g}$ be a T-S function such that $g(0), g(1) \neq \pm\infty$ and let N be the strong negation generated by g, i.e., defined as $N(t) = g^{-1}(g(0) + g(1) - g(t))$. Then $Q_{\gamma,T,S,g}$ is self-dual w.r.t. N if and only if $\gamma = 1/2$ and (T, S) are dual to each other w.r.t. N.*

  The above result can be applied, in particular, to linear convex functions $L_{\gamma,T,S}$, which appear to be self-dual w.r.t. the standard negation $N(t) = 1-t$ if and only if $\gamma = 1/2$ and T and S are dual to each other.

Idempotency T-S functions related to $T = \min$ and $S = \max$ are obviously idempotent, but these are not the only cases where T-S functions turn out to be averaging functions ([184]). For example, if $n = 2$, it is easy to check that the function $L_{1/2,T,S}$ is just the arithmetic mean whenever the pair (T, S) verifies the Frank functional equation $T(x, y)+S(x, y) = x+y$ for all $(x, y) \in [0, 1]^2$. This happens, in particular, when choosing the pair (T, S) in the Frank's families of t-norms and t-conorms (see p. 154) with the same parameter, i.e., taking $T = T^F_\lambda$ and $S = S^F_\lambda$ for some $\lambda \in [0,\infty]$.

Note 4.55. T-S functions of the form $Q_{\gamma,\min,\max,g}$ are clearly idempotent for inputs of *any* dimension, but this is not necessarily the case when non-idempotent t-norms and t-conorms are used. For example, the function $L_{1/2,T_P,S_P}$ is idempotent for $n = 2$ – the pair (T_P, S_P) verifies the Frank functional equation – but it is not idempotent for $n = 3$.

Continuity Since their generating functions are continuous and the cases where $Ran(g) = [-\infty,+\infty]$ are excluded, T-S functions are continuous if and only if their associated functions T and S are continuous.

4.5.3 Examples

Example 4.56. The linear combination of the minimum t-norm and the Łukasiewicz t-conorm (see Figure 4.16 for a 3D plot with $\gamma = 0.3$) was first used in [158]:

$$L_{\gamma,\min,S_L}(x_1,\ldots,x_n) = (1-\gamma)\min(x_1,\ldots,x_n) + \gamma\min\left(\sum_{i=1}^{n} x_i, 1\right).$$

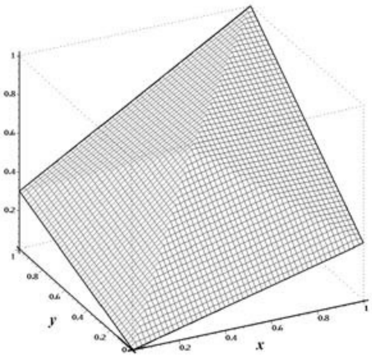


Fig. 4.16. 3D plot of $L_{0.3,\min,S_L}$ (example 4.56).

Example 4.57. Taking into account the general result on duality given in section 4.5.2, the dual function of a linear convex T-S function w.r.t. a strong negation N is a T-S function generated by $g(t) = N(t)$. T-S functions with such a generating function are expressed as:

$$Q_{\gamma,T,S,N}(x_1,\ldots,x_n) = N\Big((1-\gamma)N(T(x_1,\ldots,x_n)) + \gamma N(S(x_1,\ldots,x_n))\Big)$$

Note that if N is taken as the standard negation $N(t) = 1-t$, it is $Q_{\gamma,T,S,N} = L_{\gamma,T,S}$, i.e., $Q_{\gamma,T,S,1-Id}$ is a linear convex T-S function. Otherwise, the resulting functions are not linear combinations. To visualize a concrete example, the choice of the strong negation $N_p(t) = 1-(1-t)^p$, $p>0$ provides the following parameterized family (see Figure 4.17 for the 3D plot of a member of this family obtained with $\gamma = 0.5$, $T = T_D$, $S = \max$ and $p = 3$):

$$Q_{\gamma,T,S,N_p}(x_1,\ldots,x_n) = 1-\Big((1-\gamma)(1-T(x_1,\ldots,x_n))^p + \gamma(1-S(x_1,\ldots,x_n))^p\Big)^p$$

Example 4.58. Similarly to Example 4.57, the duals of exponential convex T-S functions w.r.t. a strong negation N are T-S functions with generating function $g(t) = \log(N(t))$. The class of T-S functions generated by $\log\circ N$ is given by:

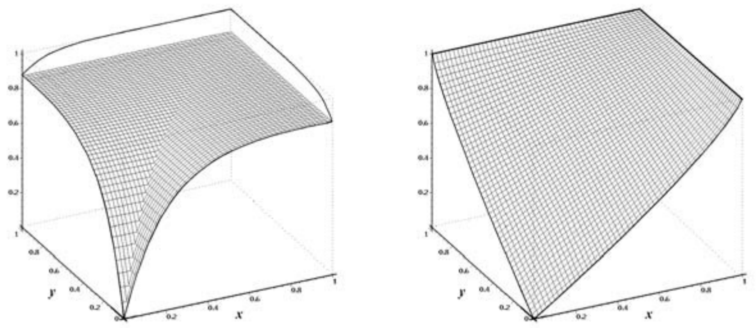


Fig. 4.17. 3D plot of $Q_{0.5,T_D,\max,N_3}$ in example 4.57 (left) and $Q_{0.8,\min,S_P,\log\circ(1-Id)}$ in example 4.58 (right).

$$Q_{\gamma,T,S,\log\circ N}(x_1,\ldots,x_n) = N\Big(N(T(x_1,\ldots,x_n))^{1-\gamma}\cdot N(S(x_1,\ldots,x_n))^{\gamma}\Big).$$

It is easy to check that these functions are never exponential convex T-S functions. A simple example is obtained with the standard negation, that provides the following function (see Figure 4.17 for the 3D plot with $\gamma = 0.8$, $T = \min$ and $S = S_P$):

$$Q_{\gamma,T,S,\log\circ(1-Id)}(x_1,\ldots,x_n) = 1-(1-T(x_1,\ldots,x_n))^{1-\gamma}\cdot(1-S(x_1,\ldots,x_n))^{\gamma}.$$

Example 4.59. Choosing $g_p(t) = t^p$, $p \in]-\infty, 0[\ \cup\ ]0, +\infty[$, the following parameterized family of T-S functions is obtained (see Figure 4.18 for a 3D plot with $\gamma = 0.5$, $T = T_P$, $S = S_L$ and $p = 2$):

$$Q_{\gamma,T,S,g_p}(x_1,\ldots,x_n) = \Big((1-\gamma)(T(x_1,\ldots,x_n))^p + \gamma(S(x_1,\ldots,x_n))^p\Big)^{1/p}.$$

Example 4.60. The choice $T = \min$ and $S = \max$ provides a wide family of idempotent T-S functions (see Figure 4.18 for a 3D plot with $\gamma = 0.2$ and $g(t) = t^3$):

$$Q_{\gamma,\min,\max,g}(x_1,\ldots,x_n) = g^{-1}\Big((1-\gamma)g(\min(x_1,\ldots,x_n)) + \gamma g(\max(x_1,\ldots,x_n))\Big).$$

4.5.4 Fitting to the data

Fitting T-S functions to a set of empirical data $\mathcal{D}$ involves: a) fitting the unknown parameter γ; b) fitting the participating t–norm and t–conorm; c) fitting the generator g; d) fitting all these parameters together. We start with the simplest case of fitting $\gamma \in [0,1]$, provided that T, S and g are given (or

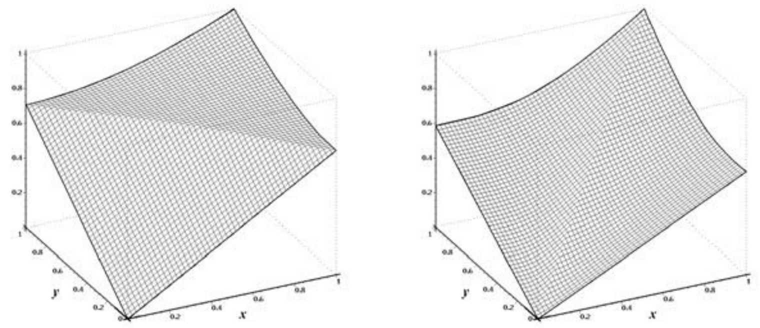


Fig. 4.18. 3D plot of the T-S function $Q_{0.5,T_P,S_L,g_2}$ in example 4.59 (left) and $Q_{0.2,\min,\max,g}$ with $g(t) = t^3$ in example 4.60 (right).

fixed). Consider first linear convex T-S functions in Definition 4.48. Let us rewrite this expression as

$$f(\mathbf{x}) = T(\mathbf{x}) + \gamma(S(\mathbf{x}) - T(\mathbf{x})).$$

Using the least squares criterion, we solve the problem

$$\text{Minimize}_{\gamma\in[0,1]} \sum_{k=1}^{K} \left(\gamma(S(\mathbf{x}_k) - T(\mathbf{x}_k)) + T(\mathbf{x}_k) - y_k\right)^2.$$

The objective function is convex, and the explicit solution is given by

$$\gamma^* = \max\left\{0, \min\left\{1, \frac{\sum\limits_{k=1}^{K}(y_k - T(\mathbf{x}_k))}{\sum\limits_{k=1}^{K}(S(\mathbf{x}_k) - T(\mathbf{x}_k))^2}\right\}\right\}. \tag{4.19}$$

For the exponential convex T-S functions, we need to linearize the outputs by taking logarithms, in which case

$$\log(f(\mathbf{x})) = \log(T(\mathbf{x})) + \gamma(\log(S(\mathbf{x})) - \log(T(\mathbf{x}))).$$

Let us denote $\tilde{T}(\mathbf{x}) = \log(T(\mathbf{x}))$, $\tilde{S}(\mathbf{x}) = \log(S(\mathbf{x}))$ and $\tilde{y} = \log(y)$. Then we have the minimization problem

$$\text{Minimize}_{\gamma\in[0,1]} \sum_{k=1}^{K} \left(\gamma(\tilde{S}(\mathbf{x}_k) - \tilde{T}(\mathbf{x}_k)) + \tilde{T}(\mathbf{x}_k) - \tilde{y}_k\right)^2$$

and the explicit solution is given by (4.19) with $\tilde{T}, \tilde{S}$ and $\tilde{y}$ replacing T, S, y.

Let now T and S be members of parametric families of t–norms and t–conorms, with unknown parameters p_T, p_S. Then minimization of the least squares criterion has to be performed with respect to three parameters p_T, p_S and γ. While for fixed p_T, p_S we have a convex optimization problem with an explicit solution (4.19), fitting parameters p_T, p_S involves a nonlinear optimization problem with potentially multiple locally optimal solutions, see Appendix A.5.5. Deterministic global optimization methods, like the Cutting Angle method or grid search, usually work well in two dimensions. Thus we will solve a bi-level optimization problem

$$\mathrm{Min}_{p_T,p_S}\mathrm{Min}_{\gamma\in[0,1]}\sum_{k=1}^{K}\left(\gamma(S_{p_S}(\mathbf{x}_k)-T_{p_T}(\mathbf{x}_k))+T_{p_T}(\mathbf{x}_k)-y_k\right)^2,$$

where at the outer level we apply a global optimization method, and at the inner level we use the explicit formula (4.19). Of course, if T and S are dual to each other, the problem simplifies significantly.

Next, let us fit the generator g, having T, S fixed, but allowing γ to vary. This problem is very similar to that of fitting generators of quasi-arithmetic means in Section 2.3.7. If g is defined by an algebraic formula with a parameter p_g, e.g., $g(t) = t^{p_g}$, then we have the following problem

$$\mathrm{Minimize}_{\gamma\in[0,1],p_g}\sum_{k=1}^{K}\left(\gamma[g_{p_g}(S(\mathbf{x}_k))-g_{p_g}(T(\mathbf{x}_k))]+g_{p_g}(T(\mathbf{x}_k))-g_{p_g}(y_k)\right)^2.$$

Again, for a fixed p_g the optimal γ can be found explicitly by appropriately modifying (4.19), while the problem with respect to p_g is nonlinear, and a global optimization method (like Pijavski-Shubert) needs to be applied.

Consider now the case when $g(t)$ is not defined parametrically, in which case we could represent it with a monotone linear regression spline

$$g(t)=\sum_{j=1}^{J}c_jB(t_j),$$

$B_j(t)$ being modified B-splines. Then, similarly to weighted quasi-arithmetic means, we have the problem of fitting spline coefficients c_j and the weight γ to data, subject to $c_j > 0$. In this case, given that the number of spline coefficients J is greater than 1, it makes sense to pose a bi-level optimization problem the other way around, i.e.,

$$\min_{\gamma\in[0,1]}\min_{c_j>0}\sum_{k=1}^{K}\left(\gamma\sum_{j=1}^{J}c_jB_j(S(\mathbf{x}_k))+(1-\gamma)\sum_{j=1}^{J}c_jB_j(T(\mathbf{x}_k))-\sum_{j=1}^{J}c_jB_j(y_k)\right)^2.$$

Next rearrange the terms of the sum to get

$$\min_{\gamma\in[0,1]} \min_{c_j>0} \sum_{k=1}^{K} \left(\sum_{j=1}^{J} c_j \left[\gamma B_j(S(\mathbf{x}_k)) + (1-\gamma) B_j(T(\mathbf{x}_k)) - B_j(y_k)\right] \right)^2 .$$

Consider the inner problem. The expression in the square brackets does not depend on c_j for a fixed γ, so we can write

$$F(\mathbf{x}_k, y_k; \gamma) = \gamma B_j(S(\mathbf{x}_k)) + (1-\gamma) B_j(T(\mathbf{x}_k)) - B_j(y_k),$$

and

$$\min_{c_j>0} \sum_{k=1}^{K} \left(\sum_{j=1}^{J} c_j F(\mathbf{x}_k, y_k; \gamma) \right)^2 .$$

This is a standard quadratic programming problem, and we solve it by QP methods, discussed in Appendix A.5. The outer problem is now a general nonlinear optimization problem, with possibly multiple local minima, and we use a univariate global optimization method, like the Pijavski-Shubert method.

Finally if we have freedom of choice of all the parameters: γ, p_T, p_S and g, then we have a global optimization problem with respect to all these parameters. It makes sense to use bi-level optimization by putting either γ or c_j (if g is represented by a spline) in the inner level, so that its solution is either found explicitly or by solving a standard QP problem.

4.6 Symmetric sums

Symmetric sums (or, more generally, N-symmetric-sums, where N is a strong negation) are nothing but self-dual (N-self-dual) aggregation functions[19].

Recall from Chapter 1 that strong negations (Definition 1.48, p. 18) may be used to construct new aggregation functions from existing ones by reversing the input scale: indeed (see Definition 1.54, p. 20), given a strong negation N, to each aggregation function $f : [0,1]^n \to [0,1]$ there corresponds another aggregation function $f_d : [0,1]^n \to [0,1]$, defined as $f_d(x_1, \dots, x_n) = N(f(N(x_1), \dots, N(x_n)))$, which is called the N-dual of f.

Recall also that the classes of conjunctive and disjunctive functions are dual to each other, that is, the N-dual of a conjunctive function is always a disjunctive one, and vice-versa. It is also easy to check[20] that the two remaining classes of aggregation functions are closed under duality, i.e., the N-dual of an averaging (respectively mixed) function is in turn an averaging

[19] The original definition in [227] included two additional axioms – continuity and symmetry – but these restrictions are normally no longer considered (see, e.g., [43]).

[20] It suffices to notice that for any strong negation N, min and max are N-dual to each other, and that $f \le g$ implies $g_d \le f_d$.

(respectively mixed) function. This means that N-self-dual aggregation functions[21] (i.e., those such that $f_d = f$, see Definition 1.55) can only be found among averaging and mixed functions. Some N-self-dual functions, belonging to the main families of averaging and mixed functions, have been identified in different sections of Chapters 2 and 4. In particular, we have:

- Weighted quasi-arithmetic means with bounded generating function g are N-self-dual with respect to the strong negation $N(t) = g^{-1}(g(0) + g(1) - g(t))$. Particularly, any weighted arithmetic mean is self-dual with respect to the standard negation (see Section 2.3.2).
- OWA functions with symmetric weighting vectors are self-dual with respect to the standard negation (Section 2.5.2).
- Nullnorms $V_{T,S,a}$, where a is the negation's fixed point (i.e., the value such that $N(a) = a$) and T and S are dual with respect to the strict negation $\tilde{N}(t) = \frac{N(t \cdot a) - a}{1-a}$. In particular, a-medians $V_{\min,\max,a}$ are N-self-dual with respect to any N with fixed point a (Section 4.3.2).
- T-S-functions $Q_{0.5,T,S,g}$ such that $N(t) = g^{-1}(g(0)+g(1)-g(t))$ and (T, S) are dual to each other w.r.t. N (Section 4.5.2).

The first studies of this kind of aggregation functions date back to 1980-s [78, 83, 190, 227], but their applications to the solution of different decision problems – such as preference modeling [102, 103] or multicriteria decision making [162]– have renewed the interest in them (see recent publications [104, 160, 161, 185, 191]).

4.6.1 Definitions

In order to define symmetric sums – N-symmetric sums – it suffices to recover the definition of self-dual aggregation function given on page 20. Let us first consider the particular case of self-duality with respect to the standard negation:

Definition 4.61 (Symmetric sum). *A symmetric sum*[22] *is an aggregation function $f : [0, 1]^n \to [0, 1]$ which is self-dual with respect to the standard negation, i.e., which verifies*

$$f(x_1, \ldots, x_n) = 1 - f(1 - x_1, \ldots, 1 - x_n)$$

for all $(x_1, \ldots, x_n) \in [0, 1]^n$.

The above definition can be easily generalized to the case of arbitrary strong negations in the following way:

---

[21] Also known as N-invariant aggregation functions.

[22] Symmetric sums are also sometimes called *reciprocal* aggregation functions [104].

Definition 4.62 (N-symmetric sum). *Given a strong negation N, a N-symmetric sum is an aggregation function $f : [0,1]^n \to [0,1]$ which is self-dual with respect to N, i.e., which verifies*

$$f(x_1, \ldots, x_n) = N\Big(f(N(x_1), \ldots, N(x_n))\Big) \tag{4.20}$$

for all $(x_1, \ldots, x_n) \in [0,1]^n$.

It is important to notice that, despite their name, N-symmetric sums **are not necessarily symmetric** in the sense of Definition 1.16. For example, weighted arithmetic means – excluding the special case of arithmetic means – are prototypical examples of non-symmetric self-dual aggregation functions.

4.6.2 Main properties

N-symmetric sums are of course characterized by the N-self-duality equation (4.20), but this equation does not provide any means for their construction. More useful characterizations, showing how to construct N-symmetric sums starting from arbitrary aggregation functions, are available. The first of these characterizations can be stated as follows[23]:

Proposition 4.63. *Let $\varphi : [0,1] \to [0,1]$ be an automorphism and let N be the strong negation generated by φ[24]. A function $f : [0,1]^n \to [0,1]$ is a N-symmetric sum if and only if there exists an aggregation function $g : [0,1]^n \to [0,1]$ such that*

$$f(x_1, \ldots, x_n) = \varphi^{-1}\left(\frac{g(x_1, \ldots, x_n)}{g(x_1, \ldots, x_n) + g(N(x_1), \ldots, N(x_n))}\right)$$

with convention $\frac{0}{0} = \frac{1}{2}$. The function g is called a generating function of f.

Proof. The sufficiency of the above characterization is obtained by choosing $g = \varphi \circ f$, and the necessity is a matter of calculation.

Note 4.64. The situation $\frac{0}{0}$ will occur if and only if $g(x_1, \ldots, x_n) = 0$ and $g(N(x_1), \ldots, N(x_n)) = 0$, and for such tuples $f(x_1, \ldots, x_n) = \varphi^{-1}(1/2)$, i.e., the value of the aggregation coincides with the negation's fixed point (see Remark 1.53). This will happen, for example, in the case of generating functions with absorbing element 0 and tuples $(x_1, \ldots, x_n)$ such that $\min(x_1, \ldots, x_n) = 0$ and $\max(x_1, \ldots, x_n) = 1$.

A characterization for standard symmetric sums is then obtained as a simple corollary of Proposition 4.63 choosing $N(t) = 1 - t$:

[23] See [83], or [227] and [43] for similar results.

[24] That is, $N(t) = N_\varphi(t) = \varphi^{-1}(1 - \varphi(t))$ (see characterization 1.51 on p. 18).

Corollary 4.65. *A function $f : [0,1]^n \to [0,1]$ is a symmetric sum if and only if there exists an aggregation function $g : [0,1]^n \to [0,1]$ such that*

$$f(x_1,\ldots,x_n) = \frac{g(x_1,\ldots,x_n)}{g(x_1,\ldots,x_n) + g(1-x_1,\ldots,1-x_n)}.$$

with convention $\frac{0}{0} = \frac{1}{2}$.

Note 4.66. When using the above characterization for building new symmetric sums, the generating function g cannot be itself self-dual, because this would imply $f = g$.

A different characterization of N-symmetric sums is the following[25]:

Proposition 4.67. *Let $\varphi : [0,1] \to [0,1]$ be an automorphism and let N be the strong negation generated by φ. A function $f : [0,1]^n \to [0,1]$ is a N-symmetric sum if and only if there exists an aggregation function $g : [0,1]^n \to [0,1]$ such that*

$$f(x_1,\ldots,x_n) = \varphi^{-1}\left(\frac{g(x_1,\ldots,x_n) + 1 - g(N(x_1),\ldots,N(x_n))}{2}\right).$$

Proof. Similar to the proof of Proposition 4.63.

Again, we obtain, as a corollary, the following method for constructing symmetric sums:

Corollary 4.68. *A function $f : [0,1]^n \to [0,1]$ is a symmetric sum if and only if there exists an aggregation function $g : [0,1]^n \to [0,1]$ such that*

$$f(x_1,\ldots,x_n) = \frac{g(x_1,\ldots,x_n) + 1 - g(1-x_1,\ldots,1-x_n)}{2}$$

Note 4.69. As it happens in the case of Corollary 4.65 (see Note 4.66), choosing a self-dual generating function g would not provide any new symmetric sum.

Note 4.70. Note that the two characterizations given in Propositions 4.63 and 4.67 are similar in that they both are of the form

$$f(x_1,\ldots,x_n) = h(g(x_1,\ldots,x_n), g_d(x_1,\ldots,x_n)) \tag{4.21}$$

that is, N-symmetric sums f are in both cases obtained as the composition (see Section 1.5) of an aggregation function g and its N-dual g_d by means of a bivariate aggregation function h. Indeed, this can be easily checked choosing

$$h(x,y) = \varphi^{-1}\left(\frac{x}{x+N(y)}\right)$$

to obtain Proposition 4.63 ($h(x,y) = \frac{x}{x+1-y}$ for Corollary 4.65) and

[25] This is just a generalization of the characterization given in [104].

$$h(x,y) = \varphi^{-1}\left(\frac{x+1-N(y)}{2}\right)$$

for Proposition 4.67 (the arithmetic mean for Corollary 4.68). Moreover, these two bivariate functions are not the only ones allowing to build N-symmetric sums from equation (4.21). Actually, any bivariate aggregation function $h : [0,1]^2 \to [0,1]$ verifying $h(x,y) = N(h(N(y), N(x)))$ and reaching every element of $[0, \varphi^{-1}(1/2)[$ is suitable [161]. A simple example of such a function h is given by the quasi-arithmetic mean generated by φ, i.e., the function

$$h(x,y) = \varphi^{-1}\left(\frac{\varphi(x)+\varphi(y)}{2}\right).$$

Note 4.71. Observe that if g is itself N-self-dual and h is idempotent, then the construction method given in equation (4.21) is not useful, since it ends up with $f = g$ (this happens, in particular, in the case of Corollaries 4.65 and 4.68, see Notes 4.66 and 4.69).

Note 4.72. The generating functions of Propositions 4.63 and 4.67 are not necessarily unique, that is, different functions g can lead to the same N-symmetric sum. A straightforward example of this appears in the case of Corollary 4.68, where any aggregation function g generates the same symmetric sum as its dual function g_d[26]. Thus, for each characterization, generating functions can be grouped into equivalence classes, each one containing all the aggregation functions that generate the same N-self-dual function[27].

Other interesting properties of N-symmetric sums are the following:

Bounds The bounds of some classes of N-symmetric sums, namely, those built as in Note 4.70, $f = h(g, g_d)$, with the additional condition of the idempotency of h, are easily calculated. Indeed, if h is idempotent, it is $\min(x,y) \le h(x,y) \le \max(x,y)$, and, consequently:

$$\min(g(\mathbf{x}), g_d(\mathbf{x})) \le f(\mathbf{x}) \le \max(g(\mathbf{x}), g_d(\mathbf{x})).$$

Note that the above inequality holds, in particular, in the cases of Corollaries 4.65 and 4.68, since both use idempotent functions h.

Symmetry Recall from Section 4.6.1 that N-symmetric sums do not need to be symmetric. Nevertheless, when $n = 2$ and f is a symmetric N-symmetric sum, the following special property holds:

$$f(t, N(t)) = t_N \quad \forall t \in [0,1],$$

where t_N is the negation's fixed point. This implies, in particular, $f(0,1) = f(1,0) = t_N$. When N is taken as the standard negation, the above equation is written as $f(t, 1-t) = \frac{1}{2}$.

[26] Note that this does not happen in the case of Corollary 4.65.

[27] For more details on this, see [161].

Absorbing element N-symmetric sums may or may not possess an absorbing element, but if they do then it must necessarily coincide with t_N, the negation's fixed point[28]. This is the case, for example, of N-self-dual nullnorms, as it was already pointed out in Section 4.3.2.

It is not difficult to explicitly build N-symmetric sums with the absorbing element t_N. For example, both Corollary 4.65 and Corollary 4.68, when used with generating functions having absorbing element $1/2$, provide symmetric sums with the same absorbing element. More generally, it is easy to check that the construction $h(g, g_d)$ mentioned in Note 4.70, when used with a generating function g with absorbing element t_N and a function h that is, in addition, idempotent, leads to an N-symmetric sum with absorbing element t_N. If φ is the generator of N, an example of this may be obtained by choosing h as the quasi-arithmetic mean generated by φ, and g as any nullnorm with absorbing element $t_N = \varphi^{-1}(1/2)$.

On the other hand, the fact that the absorbing element must necessarily coincide with t_N excludes any aggregation function with absorbing element in $\{0, 1\}$, such as: weighted quasi-arithmetic means with unbounded generating functions, e.g., the geometric and harmonic means (see Section 2.2); uninorms (Section 4.2); T-S-functions (Section 4.5) with unbounded generating functions, e.g., any gamma operator (Definition 4.46).

Neutral element Similarly to the absorbing element, if an N-symmetric sum has a neutral element, then it is the negation's fixed point. Observe that any N-symmetric sum built as in Note 4.70 by means of a generating function having neutral element t_N and an idempotent function h, has the same neutral element t_N. Examples of N-symmetric sums with neutral element built in a different way can be found below in Section 4.6.3.

Idempotency Clearly, some N-symmetric sums are idempotent (those belonging to the class of averaging functions) while others are not (the mixed ones). For example, corollaries 4.65 and 4.68 easily end up with idempotent symmetric sums as long as an idempotent generating function is used. This is not always the case with the characterizations given in Propositions 4.63 and 4.67, but it is true when using the construction $h(g, g_d)$ of Note 4.70 with g and h idempotent.

Shift-invariance Shift-invariant (Definition 1.45) symmetric sums have proved their usefulness [103]. Such functions can be built, in particular, by means of Corollary 4.68 starting from an arbitrary shift-invariant generating function g [104].

4.6.3 Examples

Some prominent classes of N-symmetric sums are presented below.

[28] The proof of this is immediate from the definition of f and the uniqueness of the fixed point of strong negations (see Remark 1.53 on p. 20).

T-norm-based N-symmetric sums

The two propositions given in the previous section – or, more generally, the construction method mentioned in Note 4.70, – along with the corresponding corollaries for the case of the standard negation, constitute powerful tools for building N-symmetric sums of different kinds and with different properties. Let us see what happens, in particular, when choosing a t-norm (Chapter 3) as a generating function [83].

Given a strong negation $N = N_\varphi$ with fixed point t_N, Proposition 4.63, choosing $g = T$ where T is a t-norm, provides the N-symmetric sum defined as

$$f(x_1, \ldots, x_n) = \varphi^{-1}\left(\frac{T(x_1, \ldots, x_n)}{T(x_1, \ldots, x_n) + T(N(x_1), \ldots, N(x_n))}\right)$$

whenever it is $T(x_1, \ldots, x_n) \neq 0$ or $T(N(x_1), \ldots, N(x_n)) \neq 0$, and

$$f(x_1, \ldots, x_n) = t_N$$

otherwise.

Example 4.73. $T = \min$ provides the following idempotent N-symmetric sum:

$$f(x_1, \ldots, x_n) = \begin{cases} \varphi^{-1}\left(\frac{\min(x_i)}{\min(x_i)+\min(N(x_i))}\right), & \text{if } \min(x_i) \neq 0 \text{ or } \max(x_i) \neq 1, \\ t_N & \text{otherwise.} \end{cases}$$

In particular, when N is the standard negation, Corollary 4.65 allows one to build t-norm-based symmetric sums given by

$$f(x_1, \ldots, x_n) = \frac{T(x_1, \ldots, x_n)}{T(x_1, \ldots, x_n) + T(1 - x_1, \ldots, 1 - x_n)}$$

whenever it is $T(x_1, \ldots, x_n) \neq 0$ or $T(1 - x_1, \ldots, 1 - x_n) \neq 0$, and

$$f(x_1, \ldots, x_n) = 1/2$$

otherwise. Observe[29] that these functions verify the inequality $T \le f \le T_d$ and, consequently, are idempotent when choosing $T = \min$.

Example 4.74. When $n = 2$, the following min-based bivariate symmetric sum is obtained (see Figure 4.19 for a 3D plot):

$$f(x, y) = \begin{cases} \frac{\min(x,y)}{1-|x-y|}, & \text{if } \{x, y\} \neq \{0, 1\}, \\ \frac{1}{2} & \text{otherwise.} \end{cases}$$

[29] See the note on bounds in the previous section.

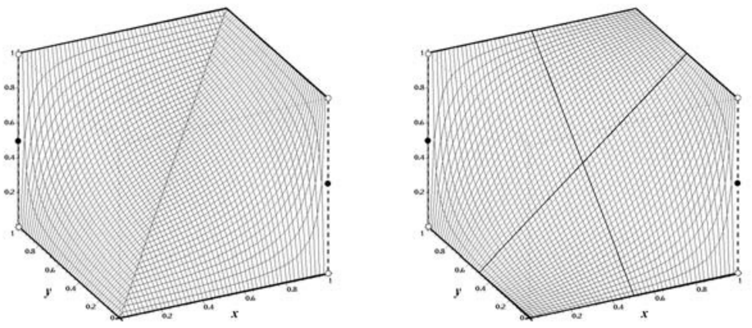


Fig. 4.19. 3D plots of the min-based idempotent symmetric sum in Example 4.74 and T_P-based symmetric sum in Example 4.75.

Example 4.75. Choosing the product t-norm T_P as the generating function in Corollary 4.65, the following symmetric sum is obtained:

$$f(x_1, x_2, \ldots, x_n) = \frac{\prod_{i=1}^{n} x_i}{\prod_{i=1}^{n} x_i + \prod_{i=1}^{n} (1 - x_i)},$$

with the convention $\frac{0}{0} = \frac{1}{2}$. Except for the tuples $(x_1, \ldots, x_n)$ such that $\{0, 1\} \subset \{x_1, \ldots, x_n\}$, that verify $f(x_1, \ldots, x_n) = \frac{1}{2}$, this function coincides with the representable uninorm known as the $3 - \Pi$ function (Example 4.19 in p. 209).

Note now that Proposition 4.67 also allows one to construct t-norm-based N-symmetric sums given by

$$f(x_1, \ldots, x_n) = \varphi^{-1}\left(\frac{T(x_1, \ldots, x_n) + 1 - T(N(x_1), \ldots, N(x_n))}{2}\right),$$

where T is an arbitrary t-norm. In the case of the standard negation (i.e., when applying Corollary 4.68) the above functions become symmetric sums of the form

$$f(x_1, \ldots, x_n) = \frac{T(x_1, \ldots, x_n) + T_d(x_1, \ldots, x_n)}{2},$$

which are nothing but the arithmetic mean of a t-norm and its dual t-conorm, that is, linear T-S-functions $L_{1/2,T,T_d}$ (see Definition 4.48). The choice $T = \min$ recovers the OWA function (see Definition 1.72) with weights $w_1 = w_n = 1/2$ and $w_i = 0$ otherwise. Recall also (see the idempotency item in Section

4.5.2) that when $n = 2$, the choices $T = \min$, $T = T_P$ and $T = T_L$ all lead to the arithmetic mean, since the dual pairs $(\min, \max)$, (T_P, S_P) and (T_L, S_L) verify Frank's functional equation $T(x, y) + S(x, y) = x + y$.

Example 4.76. Choosing the Schweizer-Sklar t-norm with parameter $\lambda = 2$ (see p. 150) the following bivariate t–norm-based symmetric sum is obtained (see Figure 4.20 for a 3D plot):

$$f(x, y) = \frac{\sqrt{\max(x^2 + y^2 - 1, 0)} + 1 - \sqrt{\max((1-x)^2 + (1-y)^2 - 1, 0)}}{2}$$

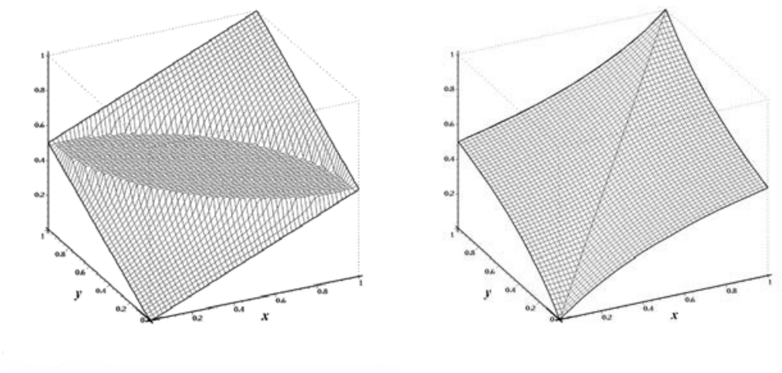


Fig. 4.20. 3D plot of the Schweizer-Sklar t–norm-based symmetric sum given in Example 4.76 (left) and of the max-based symmetric sum given in Example 4.77 (right).

T-conorm-based N-symmetric sums

The t-norms used in the previous section can of course be replaced by their dual functions, t-conorms. If $N = N_\varphi$ is a strong negation and S is an arbitrary t-conorm, Proposition 4.63 provides the N-symmetric sum defined as

$$f(x_1, \ldots, x_n) = \varphi^{-1}\left(\frac{S(x_1, \ldots, x_n)}{S(x_1, \ldots, x_n) + S(N(x_1), \ldots, N(x_n))}\right).$$

When N is taken as the standard negation, Corollary 4.65 provides t-conorm-based symmetric sums of the form

$$f(x_1, \ldots, x_n) = \frac{S(x_1, \ldots, x_n)}{S(x_1, \ldots, x_n) + S(1 - x_1, \ldots, 1 - x_n)},$$

which verify $S_d \leq f \leq S$ and are idempotent whenever $S = \max$.

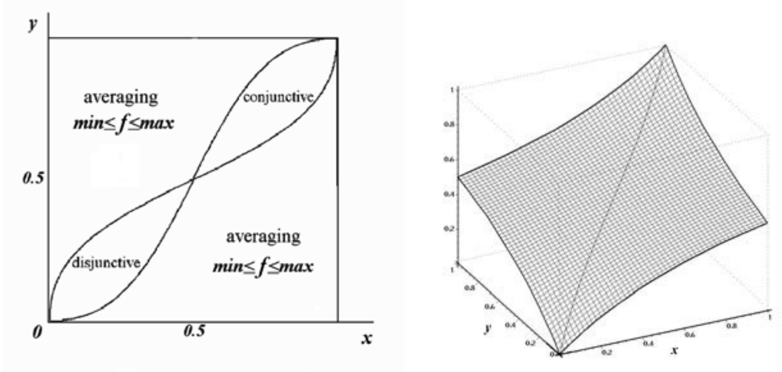


Fig. 4.21. The structure and a 3D plot of the S_P-based symmetric sum given in Example 4.77.

Example 4.77. When $n = 2$ and $N(t) = 1 - t$, the following max-based and S_P-based bivariate symmetric sums are obtained (see Figures 4.20 (right) and 4.21 for 3D plots of both functions):

$$f(x, y) = \frac{\max(x, y)}{1 + |x - y|},$$

$$f(x, y) = \frac{x + y - xy}{1 + x + y - 2xy}.$$

On the other hand, Proposition 4.67 leads to N-symmetric sums of the form

$$f(x_1, \ldots, x_n) = \varphi^{-1}\left(\frac{S(x_1, \ldots, x_n) + 1 - S(N(x_1), \ldots, N(x_n))}{2}\right).$$

When dealing with the standard negation, these functions coincide with the ones generated by their dual t-norms (see Note 4.72), that is, they are linear T-S functions.

Almost associative N-symmetric sums

If N is a strong negation with fixed point t_N and $u : [0, 1] \to [-\infty, +\infty]$ is a strictly increasing bijection, such that $u(t_N) = 0$ and verifying $u(N(t)) + u(t) = 0$ for all $t \in [0, 1]$, then the function given by

$$f(x_1, \ldots, x_n) = \begin{cases} u^{-1}\left(\sum\limits_{i=1}^{n} u(x_i)\right), & \text{if } \{0, 1\} \nsubseteq \{x_1, \ldots, x_n\}, \\ t_N & \text{otherwise} \end{cases}$$

is a N-symmetric sum with neutral element t_N which is, in addition, associative and continuous except for the tuples simultaneously containing the values 0 and 1. Note that this kind of N-symmetric sums coincide with representable uninorms (Section 4.2.3) except for the tuples $(x_1, \ldots, x_n)$ such that $\{0, 1\} \subseteq \{x_1, \ldots, x_n\}$.

4.6.4 Fitting to the data

Fitting symmetric sums to the data is based on their representation theorems (Propositions 4.63 and 4.67). Typically one has to identify the parameters of the participating aggregation function g, which could be a quasi-arithmetic means or a t–norm or t–conorm. It is a nonlinear global optimization problem, which in the case of one or two parameters can be solved using Cutting Angle method (see Appendix A.5.5).

Special classes of symmetric sums are some means, T-S functions and (up to the inputs $(x_1, \ldots, x_n)$ such that $\{0, 1\} \subseteq \{x_1, \ldots, x_n\}$) representable uninorms. Therefore special methods developed for these types of functions in Sections 2.3.7, 4.2.6 and 4.5.4 can all be applied.

4.7 ST-OWA functions

Ordered Weighted Averaging (OWA) functions (see Chapter 2) have recently been mixed with t-norms and t-conorms (Chapter 3) in order to provide new mixed aggregation functions known as T-OWAs, S-OWAs and ST-OWAs. These functions have proved to be useful, in particular, in the context of multicriteria decision making [31, 242, 273].

4.7.1 Definitions

Recall from Chapters 1 and 2 that given a weighting vector [30] $\mathbf{w} = (w_1, \ldots, w_n) \in [0, 1]^n$, an OWA is an aggregation function defined as

$$OWA_{\mathbf{w}}(x_1, \ldots, x_n) = \sum_{i=1}^{n} w_i \cdot x_{(i)} \tag{4.22}$$

where $\mathbf{x}_{()} = (x_{(1)}, \ldots, x_{(n)})$ is the vector obtained from $\mathbf{x}$ by arranging its components in non-increasing order, i.e.,$\mathbf{x}_{()} = \mathbf{x}_{\searrow}$. Due to this ordering, it is clear that for any $i \in \{1, \ldots, n\}$, $x_{(i)} = \min(x_{(1)}, \ldots, x_{(i)}) = \max(x_{(i)}, \ldots, x_{(n)})$, and then Equation (4.22) can be rewritten either as

$$OWA_{\mathbf{w}}(x_1, \ldots, x_n) = \sum_{i=1}^{n} w_i \cdot \min(x_{(1)}, \ldots, x_{(i)}), \tag{4.23}$$

[30] Definition 1.66 on p. 24.

or, equivalently, as

$$OWA_{\mathbf{w}}(x_1,\ldots,x_n)=\sum_{i=1}^{n} w_i \cdot \max(x_{(i)},\ldots,x_{(n)}). \tag{4.24}$$

Since the min function that appears in Equation (4.23) belongs to the class of t-norms (see Chapter 3), a simple generalization of OWA functions is obtained by replacing this minimum operation by an arbitrary t-norm T:

Definition 4.78 (T-OWA function). *Let $\mathbf{w} \in [0,1]^n$ be a weighting vector and let T be a t-norm*[31]*. The aggregation function $O_{T,\mathbf{w}} : [0,1]^n \to [0,1]$ defined as*

$$O_{T,\mathbf{w}}(x_1,\ldots,x_n)=\sum_{i=1}^{n} w_i \cdot T(x_{(1)},\ldots,x_{(i)}),$$

where $(x_{(1)},\ldots,x_{(n)}) = \mathbf{x}_{\searrow}$, is called a T-OWA.

Note 4.79. The following special cases should be noted:

- If $\mathbf{w} = (1,0,\ldots,0)$, then $O_{T,\mathbf{w}} = OWA_{\mathbf{w}} = \max$.
- If $\mathbf{w} = (0,\ldots,0,1)$, then $O_{T,\mathbf{w}} = T$.

Note 4.80. Bivariate T-OWA functions are just linear convex T-S functions (Definition 4.48) with $S = \max$, that is, when $n = 2$, $O_{T,\mathbf{w}} = L_{w_1,T,\max}$.

Of course, OWA functions are T-OWAs obtained when choosing $T = \min$, that is, $OWA_{\mathbf{w}} = O_{\min,\mathbf{w}}$ for any weighting vector $\mathbf{w}$. Similarly, the max function in (4.24) can be replaced by an arbitrary t-conorm S, and this results in a S-OWA:

Definition 4.81 (S-OWA function). *Let $\mathbf{w} \in [0,1]^n$ be a weighting vector and let S be a t-conorm. The aggregation function $O_{S,\mathbf{w}} : [0,1]^n \to [0,1]$ defined as*

$$O_{S,\mathbf{w}}(x_1,\ldots,x_n)=\sum_{i=1}^{n} w_i \cdot S(x_{(i)},\ldots,x_{(n)}),$$

where $(x_{(1)},\ldots,x_{(n)}) = \mathbf{x}_{\searrow}$, is called an S-OWA.

Note 4.82. The following special cases should be noted:

- If $\mathbf{w} = (1,0,\ldots,0)$, then $O_{S,\mathbf{w}} = S$.
- If $\mathbf{w} = (0,\ldots,0,1)$, then $O_{S,\mathbf{w}} = OWA_{\mathbf{w}} = \min$.

[31] Recall from Chapter 3 that t-norms are bivariate functions which are associative, and are, therefore, uniquely determined for any number of arguments (with the convention $T(t) = t$ for any $t \in [0,1]$).

Note 4.83. Similarly to the case of T-OWAs (Note 4.80), bivariate S-OWA functions are nothing but linear convex T-S functions (Definition 4.48) with $T = \min$, i.e., when $n = 2$, $O_{S,\mathbf{w}} = L_{w_1,\min,S}$.

Again, OWA functions are just particular cases of S-OWAs (obtained by choosing $S = \max$), so T-OWAs and S-OWAs both generalize OWA functions. Note, however, that they do it in opposite directions, since (see next section) T-OWAs are always weaker than OWAs whereas S-OWAs are stronger, that is, the inequality $O_{T,\mathbf{w}} \leq OWA_{\mathbf{w}} \leq O_{S,\mathbf{w}}$ holds for any weighting vector $\mathbf{w}$, any t-norm T and any t-conorm S. This observation, along with arguments rooted in the decision-making's field [245], leads to the following generalization.

Definition 4.84 (ST-OWA function). *Let $\mathbf{w} \in [0,1]^n$ be a weighting vector, σ the attitudinal character (orness measure)*[32] *of the OWA function $OWA_{\mathbf{w}}$, T a t-norm and S a t-conorm. The aggregation function $O_{S,T,\mathbf{w}} : [0,1]^n \to [0,1]$ defined as*

$$O_{S,T,\mathbf{w}}(x_1, \ldots, x_n) = \sum_{i=1}^{n} w_i \cdot \Big((1-\sigma)T(x_{(1)}, \ldots, x_{(i)}) + \sigma S(x_{(i)}, \ldots, x_{(n)})\Big),$$

where $(x_{(1)}, \ldots, x_{(n)}) = \mathbf{x}_{\searrow}$, is called an ST-OWA function.

Note 4.85. Similarly to the cases of T-OWAs and S-OWAs, the following special weighting vectors are to be noted:

- If $\mathbf{w} = (1, 0, \ldots, 0)$, then $O_{S,T,\mathbf{w}} = S$.
- If $\mathbf{w} = (0, \ldots, 0, 1)$, then $O_{S,T,\mathbf{w}} = T$.

Note 4.86. Definition 4.84 could clearly have been written as

$$O_{S,T,\mathbf{w}}(x_1, \ldots, x_n) = (1-\sigma) \cdot O_{T,\mathbf{w}}(x_1, \ldots, x_n) + \sigma \cdot O_{S,\mathbf{w}}(x_1, \ldots, x_n),$$

so ST-OWAs are just linear convex combinations of a T-OWA and a S-OWA, which become simple OWAs in the limiting case $T = \min$ and $S = \max$ (i.e., $O_{\max,\min,\mathbf{w}} = OWA_{\mathbf{w}}$).

4.7.2 Main properties

T-OWAs, S-OWAs and ST-OWAs are obviously symmetric aggregation functions. Some other interesting properties are listed below:

Comparison, Ordering and Bounds

---

[32] Recall that the orness measure of $OWA_{\mathbf{w}}$ is given (see Definition 2.2) by $orness(\mathbf{w}) = \sum_{i=1}^{n} w_i \cdot (\frac{n-i}{n-1})$.

- **T-OWAs**. For any t-norms T_1 and T_2 such that $T_1 \leq T_2$ it is $O_{T_1,\mathbf{w}} \leq O_{T_2,\mathbf{w}}$.
  This property, which is easily verified, provides the following inequalities, obtained from the basic ordering properties of t-norms (see Chapter 3):
  $$O_{T_D,\mathbf{w}} \leq O_{T_L,\mathbf{w}} \leq O_{T_P,\mathbf{w}} \leq O_{\min,\mathbf{w}} = OWA_{\mathbf{w}},$$
  $$O_{T_D,\mathbf{w}} \leq O_{T,\mathbf{w}} \leq O_{\min,\mathbf{w}} = OWA_{\mathbf{w}}.$$
  On the other hand, it is not difficult to prove that any t-norm is weaker than the T-OWA built from it, so the following lower bound may be established for any T-OWA:
  $$T \leq O_{T,\mathbf{w}}.$$
- **S-OWAs**. For any t-conorms S_1 and S_2 such that $S_1 \leq S_2$ it is $O_{S_1,\mathbf{w}} \leq O_{S_2,\mathbf{w}}$, and then:
  $$OWA_{\mathbf{w}} = O_{\max,\mathbf{w}} \leq O_{S_P,\mathbf{w}} \leq O_{S_L,\mathbf{w}} \leq O_{S_D,\mathbf{w}},$$
  $$OWA_{\mathbf{w}} = O_{\max,\mathbf{w}} \leq O_{S,\mathbf{w}} \leq O_{S_D,\mathbf{w}}.$$
  Similarly to T-OWAs, the following upper bound applies for any S-OWA:
  $$O_{S,\mathbf{w}} \leq S.$$
- **ST-OWAs**. Since ST-OWAs are linear convex combinations of their corresponding T-OWA and S-OWA functions (see Note 4.86), it is
  $$O_{T,\mathbf{w}} \leq O_{S,T,\mathbf{w}} \leq O_{S,\mathbf{w}},$$
  which of course implies
  $$T \leq O_{S,T,\mathbf{w}} \leq S.$$

Attitudinal Character (orness measure) Recall from Chapter 2, Definition 2.2 that each $OWA_{\mathbf{w}}$ has an associated number in $[0,1]$ known as the attitudinal character or orness value, which measures the distance from the minimum function, and which may be calculated using the OWA function itself as follows:

$$orness(\mathbf{w}) = OWA_{\mathbf{w}}\left(1, \frac{n-2}{n-1}, \ldots, 0\right) = \sum_{i=1}^{n} w_i \cdot \left(\frac{n-i}{n-1}\right)$$

This idea can be generalized to the case of T-OWAs, S-OWAs and ST-OWAs as follows:

- **T-OWAs**. The orness value of a T-OWA $O_{T,\mathbf{w}}$, denoted by *orness* $(T, \mathbf{w})$, is a value in $[0,1]$ given by

$$orness(T,\mathbf{w}) = O_{T,\mathbf{w}}\left(1, \frac{n-2}{n-1}, \ldots, 0\right) = \sum_{i=1}^{n} w_i \cdot T\left(1, \ldots, \frac{n-i}{n-1}\right).$$

Note that $orness(T,(1,0,\ldots,0)) = 1$ and $orness(T,(0,\ldots,0,1)) = 0$. Also, the fact that $O_{T,\mathbf{w}} \le OWA_{\mathbf{w}}$ implies $orness(T,\mathbf{w}) \le orness(\mathbf{w})$, that is, T-OWAs cannot be more disjunctive than OWA functions.

- **S-OWAs**. The orness value of a S-OWA $O_{S,\mathbf{w}}$ is a value in $[0,1]$ denoted by $orness(S,\mathbf{w})$ and given by

$$orness(S,\mathbf{w}) = O_{S,\mathbf{w}}\left(1, \frac{n-2}{n-1}, \ldots, 0\right) = \sum_{i=1}^{n} w_i \cdot S\left(\frac{n-i}{n-1}, \ldots, 0\right).$$

Similarly to the case of T-OWAs, it is $orness(S,(1,0,\ldots,0)) = 1$ and $orness(S,(0,\ldots,0,1)) = 0$, and, because of the inequality $OWA_{\mathbf{w}} \le O_{S,\mathbf{w}}$, it is $orness(\mathbf{w}) \le orness(S,\mathbf{w})$, that is, S-OWAs are at least as disjunctive as OWA functions (and, of course, more disjunctive than T-OWAs).

- **ST-OWAs**. In analogy to T-OWAs and S-OWAs, the orness value of a ST-OWA is a value $orness(S,T,\mathbf{w}) \in [0,1]$ that is computed as

$$orness(S,T,\mathbf{w}) = O_{S,T,\mathbf{w}}\left(1, \frac{n-2}{n-1}, \ldots, 0\right) = (1-\sigma) \cdot orness(T,\mathbf{w}) + \sigma \cdot orness(S,\mathbf{w}).$$

Clearly, $orness(S,T,(1,0,\ldots,0)) = 1$, $orness(S,T,(0,\ldots,0,1)) = 0$, and, in general, $orness(T,\mathbf{w}) \le orness(T,S,\mathbf{w}) \le orness(S,\mathbf{w})$.

Absorbing element We have seen in Section 4.7.1 that T-OWAs, S-OWAs and ST-OWAs coincide, in some limiting cases, with t-norms and t-conorms. This entails that, at least in these cases, they will possess an absorbing element in $\{0,1\}$. Namely:

- When $\mathbf{w} = (1,0,\ldots,0)$, since $O_{T,\mathbf{w}} = \max$ and $O_{S,\mathbf{w}} = O_{S,T,\mathbf{w}} = S$, these three functions have $a = 1$ as absorbing element.
- When $\mathbf{w} = (0,\ldots,0,1)$, the corresponding T-OWAs, S-OWAs and ST-OWAs have $a = 0$ as absorbing element, since $O_{S,\mathbf{w}} = \min$ and $O_{T,\mathbf{w}} = O_{S,T,\mathbf{w}} = T$.

On the other hand it is easy to prove that these are the only cases where T-OWAs, S-OWAs and ST-OWAs possess an absorbing element.

Continuity S-OWA, T-OWA and ST-OWA are continuous if the participating t-norm and t-conorm T and S are continuous. Furthermore, if T and S are Lipschitz (for any number of arguments), then S-OWA, T-OWA and ST-OWA are also Lipschitz with the same Lipschitz constant.

Neutral element Similarly to the case of absorbing element, T-OWAs, S-OWAs and ST-OWAs only possess a neutral element in some special limiting cases. More precisely, a T-OWA (respectively, S-OWA, ST-OWA) has a neutral element if and only if one of the following situations holds:

- $\mathbf{w} = (1, 0, \dots, 0)$, in which case it is $O_{T,\mathbf{w}} = \max$ (respectively $O_{S,\mathbf{w}} = O_{S,T,\mathbf{w}} = S$) and $e = 0$.
- $\mathbf{w} = (0, \dots, 0, 1)$, in which case it is $O_{T,\mathbf{w}} = T$ (respectively $O_{S,\mathbf{w}} = \min$, $O_{S,T,\mathbf{w}} = T$) and $e = 1$.

Duality When using the standard negation $N(t) = 1 - t$, the classes of T-OWAs and S-OWAs are dual to each other:

- The dual function of a T-OWA $O_{T,\mathbf{w}}$ is the S-OWA $O_{S_d,\widehat{\mathbf{w}}}$ where S_d is the t-conorm dual to T (i.e., $S_d(x_1, \dots, x_n) = 1-T(1-x_1, \dots, 1-x_n)$) and $\widehat{\mathbf{w}}$ is the reversed of $\mathbf{w}$, that is, $\hat{w}_j = w_{n-j+1}$ for any $j \in \{1, \dots, n\}$.
- The dual function of a S-OWA $O_{S,\mathbf{w}}$ is the T-OWA $O_{T_d,\widehat{\mathbf{w}}}$ where T_d is the t-norm dual to S and $\widehat{\mathbf{w}}$ is, as previously, the reversed of $\mathbf{w}$.

Note that the attitudinal character of a T-OWA and its dual S-OWA are complementary, that is

$$
\begin{aligned}
orness(T, \mathbf{w}) &= 1 - orness(S_d, \widehat{\mathbf{w}}), \\
orness(S, \mathbf{w}) &= 1 - orness(T_d, \widehat{\mathbf{w}}).
\end{aligned}
$$

Regarding ST-OWAs, it is easy to check that this class is closed under duality w.r.t. the standard negation, that is, the dual of a ST-OWA $O_{S,T,\mathbf{w}}$ is in turn a ST-OWA, given by $O_{S_d,T_d,\widehat{\mathbf{w}}}$. This allows one to find self-dual ST-OWAs: indeed, any ST-OWA $O_{S,T,\mathbf{w}}$ such that (T, S) is a dual pair and $\mathbf{w}$ is symmetric (i.e., it verifies $\mathbf{w} = \widehat{\mathbf{w}}$) is self-dual. The attitudinal characters of a ST-OWA and its dual are complementary, that is:

$$orness(S, T, \mathbf{w}) = 1 - orness(S_d, T_d, \widehat{\mathbf{w}})$$

The latter entails that self-dual ST-OWAs have orness value 1/2.

4.7.3 Examples

Example 4.87 (T-OWA with $T = T_L$). When using the Łukasiewicz t-norm T_L, the following T-OWA is obtained

$$O_{T_L,\mathbf{w}}(x_1, \dots, x_n) = \sum_{i=1}^{n} w_i \cdot \max\left(0, \sum_{j=1}^{i} x_{(j)} - (i-1)\right).$$

When $n = 2$ the above function may be written as follows (see Figure 4.22 for a 3D plot with weighting vector $\mathbf{w} = (0.3, 0.7)$)

$$O_{T_L,\mathbf{w}}(x,y) = w_1 \cdot \max(x,y) + (1-w_1)\cdot \max(0, x+y-1).$$

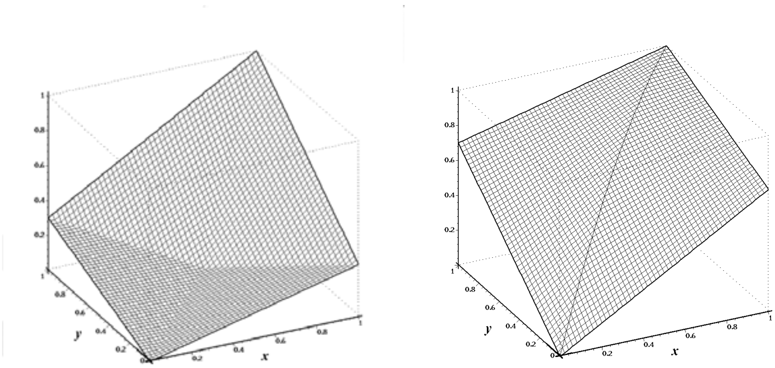


Fig. 4.22. 3D plots of the T-OWA $O_{T_L,(0.3,0.7)}$ (example 4.87) and S-OWA $O_{S_P,(0.3,0.7)}$ (example 4.88).

Example 4.88 (S-OWA with $S = S_P$). When using the probabilistic sum t-conorm S_P, the following S-OWA is obtained

$$O_{S_P,\mathbf{w}}(x_1,\ldots,x_n) = \sum_{i=1}^{n} w_i \cdot \left(1 - \prod_{j=i}^{n}(1-x_{(j)})\right).$$

When $n = 2$ the above function may be written as follows (see Figure 4.22 for a 3D plot with weighting vector $\mathbf{w} = (0.3, 0.7)$)

$$O_{S_P,\mathbf{w}}(x,y) = w_1 \cdot (x+y-xy) + (1-w_1)\cdot \min(x,y).$$

Example 4.89 (ST-OWA with $T = T_L$ and $S = S_P$). Consider now the following ST-OWA, built from the Łukasiewicz t-norm and the probabilistic sum t-conorm

$$\begin{aligned} O_{S_P,T_L,\mathbf{w}}(x_1,\ldots,x_n) &= (1-\sigma)\cdot O_{T_L,\mathbf{w}}(x_1,\ldots,x_n) + \sigma \cdot O_{S_P,\mathbf{w}}(x_1,\ldots,x_n) \\ &= (1-\sigma)\cdot \sum_{i=1}^{n} w_i \cdot \max\left(0, \sum_{j=1}^{i} x_{(j)} - (i-1)\right) \\ &\quad + \sigma \cdot \sum_{i=1}^{n} w_i \cdot \left(1 - \prod_{j=i}^{n}(1-x_{(j)})\right). \end{aligned}$$

In the bivariate case the above expression can be simplified as follows (see Figure 4.23 for a 3D plot with $\mathbf{w} = (0.3, 0.7)$)

$$O_{S_P,T_L,\mathbf{w}}(x,y) = w_1(1-w_1)(x+y) + w_1^2(x+y-xy) + (1-w_1)^2 \max(0, x+y-1).$$

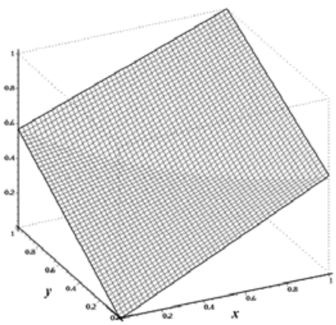


Fig. 4.23. 3D plot of the ST-OWA $O_{S_P,T_L,(0.3,0.7)}$ (example 4.89).

4.7.4 U-OWA functions

Consider now a different generalization of S- and T-OWA functions, which involves a value $e \in [0,1]$ used to separate the domain into a conjunctive, disjunctive and averaging parts, shown on Fig. 4.2 on p. 201. The purpose is to obtain a family of aggregation functions which exhibit (depending on the parameters) either averaging or conjunctive behavior on $[0,e]^n$, either averaging or disjunctive behavior on $[e,1]^n$, and averaging behavior elsewhere. The behavior of f on the subsets $[0,e]^n$ and $[e,1]^n$ is similar to that of T-OWAs and S-OWAs respectively. Our construction involves the value $e \in [0,1]$, the weighting vector $\mathbf{w}$ and scaled versions of a T-OWA and S-OWA.

We recall (see Section 4.2) that a uninorm U behaves like a scaled t-norm on $[0,e]^n$, scaled t-conorm on $[e,1]^n$ and is averaging elsewhere. U has neutral element $e \in]0,1[$ and is associative. Because of the similarity of the behavior of the function defined in Definition 4.90 with that of uninorms, we call it U-OWA. But we note that it does not possess a neutral element (except the limiting cases) nor it is associative.

Definition 4.90 (U-OWA function). *Let $\mathbf{w} \in [0,1]^n$ be a weighting vector, $e \in [0,1]$, and let T and S be a t-norm and t-conorm respectively. The aggregation function $O_{T,S,e,\mathbf{w}} : [0,1]^n \to [0,1]$ defined as*

$$O_{T,S,e,\mathbf{w}}(\mathbf{x}) = \begin{cases} \tilde{O}_{T,\mathbf{w}}(\mathbf{x}), & \text{if } \mathbf{x} \in [0,e]^n, \\ \tilde{O}_{S,\mathbf{w}}(\mathbf{x}), & \text{if } \mathbf{x} \in [e,1]^n, \\ OWA_{\mathbf{w}}(\mathbf{x}) & \text{otherwise,} \end{cases}$$

where $\tilde{O}_{T,\mathbf{w}}$ and $\tilde{O}_{S,\mathbf{w}}$ are scaled versions[33] *of T-OWA and S-OWA, is called a U-OWA.*

[33] We remind that scaled versions are obtained as $\tilde{O}_{T,\mathbf{w}}(\mathbf{x}) = e \cdot O_{T,\mathbf{w}}(\frac{x_1}{e}, \ldots, \frac{x_n}{e})$ and $\tilde{O}_{S,\mathbf{w}}(\mathbf{x}) = e + (1-e) \cdot O_{S,\mathbf{w}}(\frac{x_1 - e}{1-e}, \ldots, \frac{x_n - e}{1-e})$.

Note 4.91. The following special cases should be noted:

- If $e \in \{0, 1\}$ we obtain S-OWA and T-OWA respectively.
- If $\mathbf{w} = (1, 0, \ldots, 0)$, then $O_{T,S,e,\mathbf{w}} = (< e, 1, S >)$, an ordinal sum t-conorm, compare to Proposition 3.133.
- If $\mathbf{w} = (0, \ldots, 0, 1)$, then $O_{T,S,e,\mathbf{w}} = (< 0, e, T >)$, an ordinal sum t-norm; if $T = T_P$ then it belongs to Dubois-Prade family.

Properties of U-OWA follow from those of S-OWA and T-OWA and include

- Measure of orness $orness(T, S, e, \mathbf{w}) = O_{T,S,e,\mathbf{w}}(1, \frac{n-2}{n-1}, \ldots, 0)$;
- $orness(T, S, e, (1, 0, \ldots, 0)) = 1, orness(T, S, e, (0, 0, \ldots, 1)) = 0$;
- $O_{T,S,e,\mathbf{w}}$ has a neutral element (not coinciding with e) if and only if $\mathbf{w} = (1, 0, \ldots, 0)$ (the neutral element is 0) or $\mathbf{w} = (0, \ldots, 0, 1)$ (the neutral element is 1);
- If the underlying t-norm and t-conorm are min and max, $O_{\min,\max,e,\mathbf{w}} = OWA_{\mathbf{w}}$;
- For $n > 2$ $O_{T,S,e,\mathbf{w}}$ is continuous only in one of the limiting cases: a) $OWA_{\mathbf{w}}$, b) continuous S-OWA ($e = 0$), c) continuous T-OWA ($e = 1$), d) $\mathbf{w} = (1, 0, \ldots, 0)$ and a continuous S , e) $\mathbf{w} = (0, \ldots, 0, 1)$ and a continuous T ;
- Any U-OWA is symmetric.

4.7.5 Fitting to the data

We consider various instances of the problem of fitting parameters of ST-OWA to empirical data. As in earlier sections, we assume that there is a set of input-output pairs $\{(\mathbf{x}_k, y_k)\}, k = 1, \ldots K$, with $\mathbf{x}_k \in [0, 1]^n$, $y_k \in [0, 1]$ and n is fixed. Our goal is to determine parameters S, T, $\mathbf{w}$ which fit the data best.

Identification with fixed *S* and *T*

In this instance of the problem we assume that both S and T have been specified. The issue is to determine the weighting vector $\mathbf{w}$. For S-OWA and T-OWA, fitting the data in the least squares sense involves solution to a quadratic programming problem (QP)

$$\begin{aligned} \text{Minimize} \quad & \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i S(x_{(i)k}, \ldots, x_{(n)k}) - y_k \right)^2 \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, \end{aligned} \tag{4.25}$$

and similarly for the case of T-OWA

$$\begin{aligned}\text{Minimize} \quad & \sum_{k=1}^{K}\left(\sum_{i=1}^{n} w_i T(x_{(1)k},\ldots,x_{(i)k}) - y_k\right)^2 \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1, w_i \geq 0.\end{aligned} \tag{4.26}$$

We note that the values of S and T at any $\mathbf{x}_k$ are fixed (do not depend on $\mathbf{w}$). This problem is very similar to that of calculating the weights of standard OWA functions from data (p. 77) but involves fixed functions $S(x_{(i)k},\ldots,x_{(n)k})$ and $T(x_{(1)k},\ldots,x_{(i)k})$ rather than just $x_{(i)k}$.

If an additional requirement is to have a specified value of $orness(\mathbf{w}, S)$ and $orness(\mathbf{w}, T)$, then it becomes just an additional linear constraint, which does not change the structure of QP problem (4.25) or (4.26).

Next, consider fitting ST-OWA. Here, for a fixed value of $orness(\mathbf{w}) = \sigma$, we have a QP problem

$$\begin{aligned}\text{Minimize} \quad & \sum_{k=1}^{K}\left(\sum_{i=1}^{n} w_i ST(\mathbf{x}_k,\sigma) - y_k\right)^2 \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, \\ & orness(\mathbf{w}) = \sigma,\end{aligned} \tag{4.27}$$

where

$$ST(\mathbf{x},\sigma) = (1-\sigma)T(x_{(1)},\ldots,x_{(i)}) + \sigma S(x_{(i)},\ldots,x_{(n)}).$$

However σ may not always be specified, and hence has to be found from the data. In this case, we present a bi-level optimization problem, in which at the outer level nonlinear (possibly global) optimization is performed with respect to parameter σ, and at the inner level the problem (4.26) with a fixed σ is solved

$$\text{Minimize}_{\ \sigma\in[0,1]} \left[F(\sigma)\right], \tag{4.28}$$

where $F(\sigma)$ denotes solution to (4.27).

Numerical solution to the outer problem with just one variable can be performed by a number of methods, including grid search, multistart local search, or Pijavski-Shubert method, see Appendix A.5.5. QP problem (4.27) is solved by standard efficient algorithms, see Appendix A.5.

Identification of T-OWA and S-OWA

Consider now the problem of fitting parameters of the parametric families of participating t–norm and t–conorm, simultaneously with $\mathbf{w}$ and σ. With start with S-OWA, and assume a suitable family of t–conorms S has been chosen, e.g., Yager t–conorms S_p^Y parameterized with p. We will rely on efficient solution to problem (4.25) with a fixed S (i.e., fixed p). We set up a bi-level optimization problem

$$\text{Minimize}_{\ p\in[0,\infty]}\left[F_1(p)\right],$$

where $F_1(p)$ denotes solution to (4.25).

The outer problem is nonlinear, possibly global optimization problem, but because it has only one variable, its solution is relatively simple. We recommend Pijavski-Shubert deterministic method (Appendix A.5.5). Identification of T is performed analogously.

Next consider fitting ST-OWA functions. Here we have three parameters: the two parameters of the participating t–norm and t–conorm, which we will denote by p_1, p_2, and σ as in Problem (4.26). Of course, T and S may be chosen as dual to each other, in which case we have to fit only one parameter $p = p_1 = p_2$. To use the special structure of the problem with respect to $\mathbf{w}$ we again set up a bi-level optimization problem analogously to (4.28).

$$\text{Minimize}_{\ \sigma\in[0,1],p_1,p_2\geq 0}\left[F(\sigma,p_1,p_2)\right], \tag{4.29}$$

where $F(\sigma, p_1, p_2)$ is the solution to QP problem

$$\begin{aligned} \text{Minimize} \quad & \sum_{k=1}^{K}\left(\sum_{i=1}^{n} w_i ST(\mathbf{x}_k,\sigma,p_1,p_2) - y_k\right)^2 \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1, w_i \geq 0, \\ & orness(\mathbf{w}) = \sigma, \end{aligned} \tag{4.30}$$

and

$$ST(\mathbf{x},\sigma,p_1,p_2) = (1-\sigma)T_{p_1}^{Y}(x_{(1)},\ldots,x_{(i)}) + \sigma S_{p_2}^{Y}(x_{(i)},\ldots,x_{(n)}).$$

Solution to the outer problem is complicated because of the possibility of numerous local minima. One has to rely on methods of global optimization. One suitable deterministic method is the Cutting Angle Method (CAM) described in Appendix A.5.5.

Least absolute deviation problem

Fitting ST-OWA to the data can also be performed by using the Least Absolute Deviation (LAD) criterion [35], by replacing the sum of squares in (4.25) and (4.26) with the sum of absolute values. As described in Appendix A.2, such a problem is converted to linear programming, which replaces QP problems (4.25) – (4.27), (4.30). The outer nonlinear optimization problems do not change.

4.8 Key references

T. Calvo, A. Kolesárová, M. Komorníková, and R. Mesiar. Aggregation operators: properties, classes and construction methods. In T. Calvo, G. Mayor, and R. Mesiar, editors, *Aggregation Operators. New Trends and Applications*, pages 3–104. Physica-Verlag, Heidelberg, New York, 2002.

T. Calvo, B. De Baets, and J. Fodor. The functional equations of Frank and Alsina for uninorms and nullnorms. *Fuzzy Sets and Systems*, 120:385–394, 2001.

D. Dubois and H. Prade. A review of fuzzy set aggregation connectives. *Inform. Sci.*, 36:85–121, 1985.

E.P. Klement, R. Mesiar, and E. Pap. *Triangular Norms*. Kluwer, Dordrecht, 2000.

M. Komorníková. Aggregation operators and additive generators. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9:205–215, 2001.

W. Silvert. Symmetric summation: A class of operations on fuzzy sets. *IEEE Trans. on Systems, Man and Cybernetics*, 9:659–667, 1979.

R. Yager and D. Filev. *Essentials of Fuzzy Modelling and Control*. J. Wiley & Sons, New York, 1994.

R. Yager and A. Rybalov. Uninorm aggregation operators. *Fuzzy Sets and Systems*, 80:111–120, 1996.

R. Yager. Extending multicriteria decision making by mixing t-norms and OWA operators. *Int. J. of General Systems*, 20:453–474, 2005.

H.-J. Zimmermann and P. Zysno. Latent connectives in human decision making. *Fuzzy Sets and Systems*, 4:37–51, 1980.

5

Choice and Construction of Aggregation Functions

5.1 Problem formalization

In the previous Chapters we have studied many different families of aggregation functions, that have been classified into four major classes: conjunctive, disjunctive, averaging and mixed aggregation functions. For applications it is very important to choose the right aggregation function, or maybe a number of them, to properly model the desired behavior of the system. There is no single solution, in each application the aggregation functions are different.

The usual approach is to choose a sufficiently flexible family of aggregation functions, and then adapt it to some sort of empirical data, be it observed or desired data. It is also important to satisfy application specific requirements, such as having a neutral element, symmetry or idempotency.

In Chapters 2-4, when discussing different classes of aggregation functions, we considered specific methods of fitting a given class or family of functions to empirical data. There are many similarities in the methods we considered, but also important differences. Now we present a unifying approach to selecting an aggregation function based on empirical data.

We reiterate that typically the data comes in pairs $(\mathbf{x}, y)$, where $\mathbf{x} \in [0,1]^n$ is the input vector and $y \in [0,1]$ is the desired output. There are several pairs, which will be denoted by a subscript k: $(\mathbf{x}_k, y_k), k = 1, \ldots, K$. The data may be the result of a mental experiment: if we take the input values (x_1, x_2, x_3), what output do we expect? The data can be observed in a controlled experiment: the developer of an application could ask the domain experts to provide their opinion on the desired outputs for selected inputs. The data can also be collected in another sort of experiment, by asking a group of lay people or experts about their input and output values, but without associating these values with some aggregation rule. Finally, the data can be collected automatically by observing the responses of subjects to various stimuli. For example, by presenting a user of a computer system with some information and recording their actions or decisions.

Typical characteristics of the data set are: a) some components of vectors $\mathbf{x}_k$ may be missing; b) vectors $\mathbf{x}_k$ may have varying dimension by construction; c) the outputs y_k could be specified as a range of values (i.e., the interval $[\underline{y}_k, \overline{y}_k]$); d) the data may contain random noise; e) the abscissae of the data are scattered.

We now formalize the selection problem.

Problem 5.1 (Selection of an aggregation function). Let us have a number of mathematical properties $\mathcal{P}_1, \mathcal{P}_2, \ldots$ and the data $\mathcal{D} = \{(\mathbf{x}_k, y_k)\}_{k=1}^K$. Choose an aggregation function f consistent with $\mathcal{P}_1, \mathcal{P}_2, \ldots$, and satisfying $f(\mathbf{x}_k) \approx y_k, k = 1, \ldots, K$.

5.2 Fitting empirical data

In this section we will discuss different interpretations of the requirement $f(\mathbf{x}_k) \approx y_k, k = 1, \ldots, K$, which will lead to different types of regression problems.

The most common interpretation of the approximate equalities in Problem 5.1 is in the least squares sense, which means that the goal is to minimize the sum of squares of the differences between the predicted and observed values. By using the notion of residuals $r_k = f(\mathbf{x}_k) - y_k$, the regression problem becomes

$$\begin{aligned} &\text{minimize } ||\mathbf{r}||_2^2 \\ &\text{subject to } f \text{ satisfying } \ \mathcal{P}_1, \mathcal{P}_2, \ldots, \end{aligned} \tag{5.1}$$

where $||\mathbf{r}||_2$ is the Euclidean norm of the vector of residuals [1].

Another useful interpretation is in the least absolute deviation sense, in which the $||\cdot||_1$ norm of the residuals is minimized, i.e.,

$$\begin{aligned} &\text{minimize } ||\mathbf{r}||_1 \\ &\text{subject to } f \text{ satisfying } \ \mathcal{P}_1, \mathcal{P}_2, \ldots. \end{aligned} \tag{5.2}$$

Of course, it is possible to choose any p-norm, Chebyshev norm, or any other common criterion, such as Maximal Likelihood (ML). Also, if the data have different degrees of importance, or accuracy, then weighted analogues of the mentioned criteria are used.

A different interpretation of the approximate equality conditions was suggested in [138]. It was asserted that when empirical data comes from human judgements, the actual numerical values of y_k are not as important as the order in which the outputs are ranked. Thus one expects that if $y_k \le y_l$, then it should be $f(\mathbf{x}_k) \le f(\mathbf{x}_l)$ for all l, k. Indeed, people are better at ranking the alternatives than at assigning consistent numerical values. It is also a common

[1] See p. 22 for definitions of various norms.

practice to ask people to rank the alternatives in order of decreasing preferences (for example, during elections). Thus Problem 5.1 can be viewed as that of satisfying linear constraints $f(\mathbf{x}_k) \leq f(\mathbf{x}_l)$ if $y_k \leq y_l$ for all pairs k, l.

It is also conceivable to consider a mixture of both approaches: to fit the numerical data and to preserve the ranking. This is a two-criteria optimization problem, and to develop a method of solution, we will aggregate both criteria in some way.

5.3 General problem formulation

Many types of aggregation functions can be represented in a generic form

$$y = f(\mathbf{x}; \mathbf{w}),$$

where $\mathbf{w}$ is a vector of parameters. For example, $\mathbf{w}$ is a vector of weights for weighted means and OWA functions, or $\mathbf{w} = (\lambda)$ is a single parameter of a parametric family of t–norms or power means, or $\mathbf{w} = (\lambda_T, \lambda_S, \alpha)$ is a vector of parameters of a uninorm (p. 211), or $\mathbf{w} = \mathbf{c}$ is the vector of spline coefficients (p. 173). Moreover, in many interesting cases f depends on $\mathbf{w}$ *linearly*, and we will make use of it in the next Section. The properties $\mathcal{P}_1, \mathcal{P}_2, \ldots$ are expressed through the conditions on $\mathbf{w}$.

Least squares fitting

The generic optimization problem is

$$\text{Minimize} \sum_{k=1}^{K} (f(\mathbf{x}_k; \mathbf{w}) - y_k)^2,$$

$$\text{subject to (s.t.) conditions on } \mathbf{w}.$$

Least absolute deviation fitting

The generic optimization problem is

$$\text{Minimize} \sum_{k=1}^{K} |f(\mathbf{x}_k; \mathbf{w}) - y_k|,$$

$$\text{subject to conditions on } \mathbf{w}.$$

Preservation of output rankings

To preserve the ranking of the outputs, we re-order the data in such a way that $y_1 \leq y_2 \leq \ldots \leq y_K$ (this can always be done by sorting). The condition for order preservation takes the form

$$f(\mathbf{x}_k; \mathbf{w}) \leq f(\mathbf{x}_{k+1}; \mathbf{w}), \text{ for all } k = 1, \ldots, K-1. \tag{5.3}$$

Then condition (5.3) is added to the least squares (LS) and the least absolute deviation (LAD) problem in the form of constraints.

Fitting data and output rankings

Since empirical data have an associated noise, it may be impossible to satisfy all the constraints on $\mathbf{w}$ by using a specified class of aggregation functions. The system of constraints is said to be inconsistent.

Consider a revised version of the least squares problem

$$\begin{aligned} \text{Minimize} \quad & \sum_{k=1}^{K} (f(\mathbf{x}_k, \mathbf{w}) - y_k)^2 + \\ & P \sum_{k=1}^{K-1} \max\{f(\mathbf{x}_k, \mathbf{w}) - f(\mathbf{x}_{k+1}, \mathbf{w}), 0\}, \\ \text{s.t.} \quad & \text{other conditions on } \mathbf{w}. \end{aligned} \tag{5.4}$$

Here P is the penalty parameter: for small values of P we emphasize fitting the numerical data, while for large values of P we emphasize preservation of ordering. Of course, the second sum may not be zero at the optimum, which indicates inconsistency of constraints.

5.4 Special cases

In their general form, problems in the previous section do not have a special structure, and their solution involves a difficult nonlinear global optimization problem. While there are a number of generic tools to deal with such problems, they are only useful for small numbers of parameters. In this section we formulate the mentioned problems in the special case when f depends on the vector of parameters $\mathbf{w}$ *linearly*, which suits a large class of aggregation functions. In this case, the optimization problems do have a special structure: they become either quadratic or linear programming problems, and the solution is performed by very efficient numerical methods.

Thus we assume that

$$f(\mathbf{x}; \mathbf{w}) = < \mathbf{g}(\mathbf{x}), \mathbf{w} > = \sum_{i=1}^{n} w_i g_i(\mathbf{x}),$$

$\mathbf{g}$ being a vector of some basis functions.

Example 5.2 (Weighted arithmetic means (Section 2.2)). In this case $\mathbf{w}$ is the weighting vector and

$$g_i(\mathbf{x}) = x_i.$$

The usual constraints are $w_i \geq 0$, $\sum_{i=1}^n w_i = 1$.

Example 5.3 (Ordered Weighted Averaging (Section 2.5)). In this case $\mathbf{w}$ is the weighting vector and

$$g_i(\mathbf{x}) = x_{(i)},$$

where $x_{(i)}$ denotes the i-th largest component of $\mathbf{x}$. The usual constraints are $w_i \geq 0$, $\sum_{i=1}^n w_i = 1$. In addition, a frequent requirement is a fixed orness measure,

$$< \mathbf{a}, \mathbf{w} >= \alpha,$$

where $a_i = \frac{n-i}{n-1}$, and $0 \leq \alpha \leq 1$ is specified by the user.

Example 5.4 (Choquet integrals (Section 2.6)). We remind that a fuzzy measure is a monotonic set function $v : 2^{\mathcal{N}} \to [0,1]$ which satisfies $v(\emptyset) = 0, v(\mathcal{N}) = 1$. The discrete Choquet integral is

$$C_v(\mathbf{x}) = \sum_{i=1}^{n} \left[x_{(i)} - x_{(i-1)}\right] v(H_i), \tag{5.5}$$

where $x_{(0)} = 0$ by convention, and $H_i = \{(i), \dots, (n)\}$ is the subset of indices of $n - i + 1$ largest components of $\mathbf{x}$. A fuzzy measure has 2^n values, two of which are fixed $v(\emptyset) = 0$, $v(\mathcal{N}) = 1$.

As it was discussed in Section 2.6.6, we represent Choquet integral as a scalar product $< \mathbf{g}(\mathbf{x}), \mathbf{v} >$, where $\mathbf{v} \in [0,1]^{2^n}$ is the vector of values of the fuzzy measure. It is convenient to use the index $j = 0, \dots, 2^n - 1$ whose binary representation corresponds to the characteristic vector of the set $\mathcal{J} \subseteq \mathcal{N}$, $\mathbf{c} \in \{0,1\}^n$ defined by $c_{n-i+1} = 1$ if $i \in \mathcal{J}$ and 0 otherwise. For example, let $n = 5$; for $j = 101$ (binary), $\mathbf{c} = (0,0,1,0,1)$ and $v_j = v(\{1,3\})$. We shall use letters $\mathcal{K}, \mathcal{J}$, etc., to denote subsets that correspond to indices k, j, etc.

Define the basis functions $g_j, j = 0, \dots, 2^n - 1$ as

$$g_j(\mathbf{x}) = \max(0, \min_{i \in \mathcal{J}} x_i - \max_{i \in \mathcal{N} \setminus \mathcal{J}} x_i),$$

where $\mathcal{J} \subseteq \mathcal{N}$ is the subset whose characteristic vector corresponds to the binary representation of j. Then

$$C_v(\mathbf{x}) =< \mathbf{g}(\mathbf{x}), \mathbf{v} > .$$

Example 5.5 (Linear T-S functions (Section 4.5)).

$$f(\mathbf{x}; \mathbf{w}) = w_1 T(\mathbf{x}) + w_2 S(\mathbf{x}) =< \mathbf{g}(\mathbf{x}), \mathbf{w} >,$$

$\mathbf{g} = (T, S)$, $w_1 + w_2 = 1, w_1, w_2 \geq 0$.

Least squares fitting

$$\begin{aligned} \text{Minimize} \quad & \sum_{k=1}^{K} (< \mathbf{g}(\mathbf{x}_k), \mathbf{w} > -y_k)^2, \\ \text{s.t.} \quad & \text{linear conditions on } \mathbf{w}. \end{aligned} \tag{5.6}$$

Note that this is a standard quadratic programming problem due to the positive semidefinite quadratic objective function and linear constraints.

Least absolute deviation fitting

By using the auxiliary variables $r_k^+, r_k^- \geq 0$, such that $r_k^+ + r_k^- = | < \mathbf{g}(\mathbf{x}_k), \mathbf{w} > -y_k|$, and $r_k^+ - r_k^- =< \mathbf{g}(\mathbf{x}_k), \mathbf{w} > -y_k$, we convert the LAD problem into a linear programming problem

$$\begin{aligned} \text{Minimize} \quad & \sum_{k=1}^{K} r_k^+ + r_k^-, \\ \text{s.t.} \quad & < \mathbf{g}(\mathbf{x}_k), \mathbf{w} > -r_k^+ + r_k^- = y_k, \\ & k = 1, \dots, K, \\ & r_k^+, r_k^- \geq 0, k = 1, \dots, K, \\ & \text{other linear conditions on } \mathbf{w}. \end{aligned} \tag{5.7}$$

Least squares with preservation of output rankings

$$\begin{aligned} \text{Minimize} \quad & \sum_{k=1}^{K} (< \mathbf{g}(\mathbf{x}_k), \mathbf{w} > -y_k)^2, \\ \text{s.t.} \quad & < \mathbf{g}(\mathbf{x}_{k+1}), \mathbf{w} > - < \mathbf{g}(\mathbf{x}_k), \mathbf{w} > \geq 0, \\ & k = 1, \dots, K-1, \\ & \text{other linear conditions on } \mathbf{w}. \end{aligned} \tag{5.8}$$

This is also a standard quadratic programming problem which differs from (5.6) only by $K-1$ linear constraints.

Least absolute deviation with preservation of output rankings

$$\begin{aligned}
\text{Minimize} \quad & \sum_{k=1}^{K} r_k^+ + r_k^-, \qquad (5.9)\\
\text{s.t.} \quad & < \mathbf{g}(\mathbf{x}_k), \mathbf{w} > -r_k^+ + r_k^- = y_k,\\
& k = 1, \dots, K,\\
& < \mathbf{g}(\mathbf{x}_{k+1}) - \mathbf{g}(\mathbf{x}_k), \mathbf{w} > \geq 0,\\
& k = 1, \dots, K-1,\\
& r_k^+, r_k^- \geq 0, k = 1, \dots, K,\\
& \text{other linear conditions on } \mathbf{w}.
\end{aligned}$$

This is also a standard linear programming problem which differs from (5.7) only by $K-1$ linear constraints.

Fitting data and output rankings

$$\begin{aligned}
\text{Minimize} \quad & \sum_{k=1}^{K} (< \mathbf{g}(\mathbf{x}_k), \mathbf{w} > -y_k)^2 + \qquad (5.10)\\
& P \sum_{k=1}^{K-1} \max\{< \mathbf{g}(\mathbf{x}_k) - \mathbf{g}(\mathbf{x}_{k+1}), \mathbf{w} >, 0\},\\
\text{s.t.} \quad & \text{other linear conditions on } \mathbf{w}.
\end{aligned}$$

This is no longer a quadratic programming problem. It is a nonsmooth but convex optimization problem, and there are efficient numerical methods for its solution [9, 139, 154, 205].

However, for the LAD criterion, we can preserve the structure of the LP, namely we convert Problem (5.9) into an LP problem using auxiliary variables q_k

$$\begin{aligned}
\text{Minimize} \quad & \sum_{k=1}^{K} r_k^+ + r_k^- + P \sum_{k=1}^{K-1} q_k \qquad (5.11)\\
\text{s.t.} \quad & < \mathbf{g}(\mathbf{x}_k), \mathbf{w} > -r_k^+ + r_k^- = y_k,\\
& k = 1, \dots, K,\\
& q_k + < \mathbf{g}(\mathbf{x}_{k+1}) - \mathbf{g}(\mathbf{x}_k), \mathbf{w} > \geq 0,\\
& k = 1, \dots, K-1,\\
& q_k, r_k^+, r_k^- \geq 0, k = 1, \dots, K,\\
& \text{other linear conditions on } \mathbf{w}.
\end{aligned}$$

5.4.1 Linearization of outputs

It is possible to treat weighted quasi-arithmetic means, generalized OWA, generalized Choquet integrals and t–norms and t–conorms in the same framework, by linearizing the outputs.

Let $h : [0,1] \to [-\infty, \infty]$, be a given continuous strictly monotone function. A weighted quasi-arithmetic mean (see Section 2.3) is the function

$$f(\mathbf{x};\mathbf{w}) = h^{-1}(\sum_{i=1}^{n} w_i h(x_i)).$$

By applying h to the outputs y_k we get a quadratic programming problem which incorporates preservation of output rankings

$$\begin{aligned} \text{Minimize} \quad & \sum_{k=1}^{K} (< h(\mathbf{x}_k), \mathbf{w} > -h(y_k))^2, \\ \text{s.t.} \quad & < h(\mathbf{x}_{k+1}) - h(\mathbf{x}_k), \mathbf{w}) >\geq 0, \\ & k = 1, \dots, K-1, \\ & \sum w_i = 1, w_i \geq 0, \end{aligned} \tag{5.12}$$

where $h(\mathbf{x}) = (h(x_1), \dots, h(x_n))$. In the very same way we treat generalized OWA and generalized Choquet integrals.

For t–norms and t–conorms, we shall use the method based on fitting their additive generators, discussed in Section 3.4.15. We write the additive generator in the form of regression spline

$$S(t) =< \mathbf{B}(t), \mathbf{c} >,$$

where $\mathbf{B}(t)$ is a vector of B-splines, and $\mathbf{c}$ is the vector of spline coefficients. The conditions of monotonicity of S are imposed through linear restrictions on spline coefficients, and the additional conditions $S(1) = 0, S(0.5) = 1$ also translate into linear equality constraints.

After applying S to the output y_k we obtain the least squares problem

$$\begin{aligned} \text{Minimize} \quad & \sum_{k=1}^{K} (\sum_{i=1}^{n} < \mathbf{B}(x_{kn}), \mathbf{c} > - < \mathbf{B}(y_k), \mathbf{c} >)^2 \\ \text{s.t.} \quad & \text{linear restrictions on } \mathbf{c}. \end{aligned} \tag{5.13}$$

By rearranging the terms of the sum we get a QP problem

$$\begin{aligned} \text{Minimize} \quad & \sum_{k=1}^{K} (< \left[\sum_{i=1}^{n} \mathbf{B}(x_{kn}) - \mathbf{B}(y_k) \right], \mathbf{c} >)^2 \\ \text{s.t.} \quad & \text{linear restrictions on } \mathbf{c}. \end{aligned} \tag{5.14}$$

Next we add preservation of outputs ordering conditions, to obtain the following QP problem (note that the sign of inequality has changed because S is decreasing)

$$\text{Minimize} \qquad \sum_{k=1}^{K} \left(< \left[\sum_{i=1}^{n} \mathbf{B}(x_{kn}) - \mathbf{B}(y_k) \right], \mathbf{c} >\right)^2 \tag{5.15}$$

$$\text{s.t.} < \left[\sum_{i=1}^{n} \mathbf{B}(x_{k+1,n}) - \sum_{i=1}^{n} \mathbf{B}(x_{k,n}) \right], \mathbf{c} > \leq 0,$$

$$k = 1, \dots, K-1,$$

$$\text{linear restrictions on } \mathbf{c}.$$

LAD criterion results similarly in an LP problem.

We see that various classes of aggregation functions allow a representation in which they depend on the vector of parameters linearly, possibly after linearization. The problems of least squares, least absolute deviation, preservation of output ordering and their combinations can all be treated in the same framework, and reduced to either standard quadratic or linear programming, which greatly facilitates their solution.

5.5 Assessment of suitability

When fitting aggregation functions to numerical data, it is also important to exclude various biases, due to the chosen class or family of functions. These biases can be due to inappropriate choice of family, or inconsistency of its properties with the data. We examine two useful tools which can be used for *a posteriori* analysis of the constructed aggregation function.

The first method is subjective. It consists in plotting the predicted values of the outputs against their observed values, as on Fig. 5.1. If the estimate f is unbiased, then the dots should lie along the diagonal of the square, and on the diagonal for the data that are interpolated. How far the dots are from the diagonal reflects the accuracy of approximation. However, ideally the dots should be spread evenly above and below the diagonal. If this is not the case, say the dots are grouped below the diagonal, this is a sign of systematic underestimation of y_k, i.e., a bias. Similarly, dots above the diagonal mean systematic overestimation.

For example, let the data be consistent with the averaging type aggregation, and let us use a conjunctive aggregation function, like a t–norm, to fit it. Clearly, the predicted output values are bounded from above by the max function, and any t–norm will systematically underestimate the outputs. Based on this plot, the user can visually detect whether a chosen class of aggregation functions is really suitable for given data.

It is also possible to label the dots to see in which specific regions the constructed aggregation function exhibits bias (e.g., only for small or only for large values of y_k).

The second method is quantitative, it is based on statistical tests [286]. The simplest such test is to compute the correlation coefficients between the predicted and observed values. For example, if correlation is higher than 0.95, then the aggregation function models the data well.

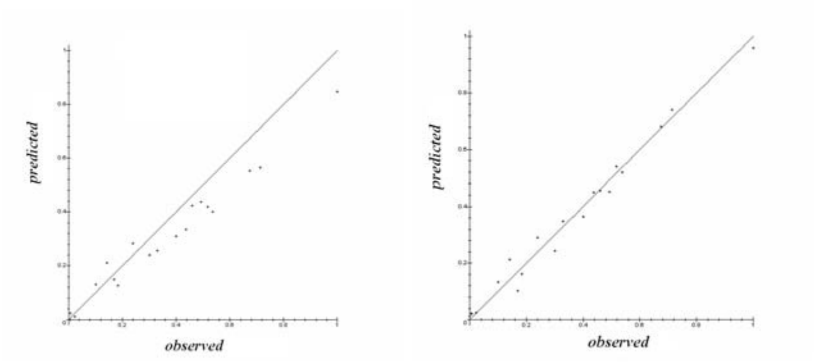


Fig. 5.1. Plots of predicted versus observed values. The plot on the left shows a bias (systematic underestimation), the plot on the right shows no bias.

The second test computes the probability that the mean difference of the predicted and observed values is not different from zero. Specifically, this test, known as one population mean test, computes the mean of the differences of the predicted and observed values $m_K = \frac{1}{K}\sum_{k=1}^{K}(f(\mathbf{x}_k) - y_k)$ of the sample, and compares it to 0. Of course, m_K is a random variable, which should be normally distributed around the mean of the population. The null hypothesis is that the true mean (of the population) is indeed zero, and the observed value of m_K (of the sample) is different from zero due to finite sample size. The alternative hypothesis is that the mean of the population is not zero, and there is a systematic bias. The Student's (two-tailed) t–test can be used. It requires the mean of the sample, the observed standard deviation s and the sample size K. The statistical test provides the probability (the P-value) of transition, i.e., the probability of observing the value of test statistic at least as large as the value actually observed (i.e., $\frac{m_K}{s/\sqrt{K}}$), if the null hypothesis were true. Small P-values indicate that the null hypothesis (the mean difference of the population is zero) is very unlikely.

Example 5.6. [286] Experimental data y_k were compared to the values of the min aggregation function. $K = 20$ is the sample size, and the computed mean of the sample was $m_K = 0.052$ with the standard deviation $s = 0.067$ (the data are plotted on Fig. 5.1 (left)). The test statistic is $t = 3.471$, with $df = 19$ (degrees of freedom), the P-value was less than 0.01. We deduce that the observed value of t will occur with probability less than 0.01 if the actual mean of the population were 0. Hence we reject the null hypothesis at 1% significance level, and conclude that min does not provide an unbiased estimate of the aggregation function used by the subjects of the experiment.

6

Interpolatory Type Aggregation Functions

6.1 Semantics

Many classes of aggregation functions we have considered so far provide a large arsenal of tools to be used in specific applications. The parameters of various families and the vectors of weights can be adjusted to fit numerical data. Yet for certain applications these classes are not flexible enough to be fully consistent with the desired inputs-outputs. Sometimes the problem specification is not sufficient to make a call as to what type of aggregation function must be used.

In this Chapter we consider an alternative construction, which is based almost entirely on the empirical data, but also incorporates important application specific properties if required. The resulting aggregation function does not belong to any specific family: it is a general aggregation function, which, according to Definition 1.5 is simply a monotone non-decreasing function satisfying $f(\mathbf{0}) = 0$ and $f(\mathbf{1}) = 1$. Furthermore, this function is not given by a simple algebraic formula, but typically as an algorithm. Yet it is an aggregation function, and for computational purposes, which is the main use of aggregation functions, it is as good as an algebraic formula in terms of efficiency.

Of course, having such a "black-box" function is not as transparent to the user of a system, nor is it easy to replicate calculations with pen and paper. Still many such black-box functions are routinely used in practice, for example neural networks for pattern recognition.

What is important, though, is that the outputs of such an aggregation function are always consistent with the data and the required properties. This is not easy to achieve using standard off-the-shelf tools, such as neural network libraries. The issue here is that no application specific properties are incorporated into the construction process, and as a consequence, the solution may fail to satisfy them (even if the data does). For example, the function determined by a neural network may fail to be monotone, hence it is not an aggregation function.

In this Chapter we present a number of tools that do incorporate monotonicity and other conditions. They are based on methods of monotone multivariate interpolation and approximation, which are outlined in the Appendix A.4. The resulting functions are of interpolatory type: they are based on interpolation of empirical data. Interpolatory constructions are very recent, they have been studied in [15, 17, 19, 25, 26, 27, 67, 109, 140, 150, 197].

The construction is formulated as the following problem. Given a set (possibly uncountable) of values of an aggregation function f, construct f, subject to a number of properties, which we discuss later in this Chapter. The pointwise construction method results in an algorithm whose output is a value of the aggregation function f at a given point $\mathbf{x} \in [0, 1]^n$.

Continuous and Lipschitz-continuous aggregation functions are of our particular interest. Lipschitz aggregation functions are very important for applications, because small errors in the inputs do not drastically affect the behavior of the system. The concept of p-stable aggregation functions was proposed in [45]. These are precisely Lipschitz continuous aggregation functions whose Lipschitz constant M in $||\cdot||_p$ norm is one [1].

The key parts of our approach are monotone interpolation techniques. We consider two methods: a method based on tensor-product monotone splines and a method based on Lipschitz interpolation. The latter method is specifically suitable for p-stable aggregation functions, and it also delivers the strongest, the weakest and the optimal aggregation functions with specified conditions.

We now proceed with the mathematical formulation of the problem. Given a data set $\mathcal{D} = \{(\mathbf{x}_k, y_k)\}_{k=1}^K, \mathbf{x}_k \in [0, 1]^n, y_k \in [0, 1]$, and a number of properties outlined below, construct an aggregation function f, such that $f(\mathbf{x}_k) = y_k$ and all the properties are satisfied. The mentioned properties of an aggregation function define a class of functions $\mathcal{F}$, typically consisting of more than just one function. Our goal is to ensure that $f \in \mathcal{F}$, and if possible, that f is in some sense the "best" element of $\mathcal{F}$.

6.2 Construction based on spline functions

6.2.1 Problem formulation

Monotone tensor product splines are defined as

$$f(x_1, \ldots, x_n) = \sum_{j_1=1}^{J_1} \sum_{j_2=1}^{J_2} \ldots \sum_{j_n=1}^{J_n} c_{j_1 j_2 \ldots j_n} B_{j_1}(x_1) B_{j_2}(x_2) \ldots B_{j_n}(x_n). \quad (6.1)$$

The univariate basis functions are chosen to be linear combinations of standard B-splines, as in [13, 15, 32], which ensures that the conditions of

[1] See p. 22 for definitions of various norms.

monotonicity of f are expressed as linear conditions on spline coefficients $c_{j_1 j_2 \ldots j_n}$.

The computation of spline coefficients (there are $J_1 \times J_2 \times \ldots \times J_n$ of them, where J_i is the number of basis functions in respect to each variable) is performed by solving a quadratic programming problem

$$\text{minimize} \sum_{k=1}^{K} \left(\sum_{j_1=1}^{J_1} \ldots \sum_{j_n=1}^{J_n} c_{j_1 j_2 \ldots j_n} B_{j_1}(x_{1k}) \ldots B_{j_n}(x_{nk}) - y_k \right)^2, \qquad (6.2)$$

subject to

$$\sum_{j_1=1}^{J_1} \ldots \sum_{j_n=1}^{J_n} c_{j_1 j_2 \ldots j_n} \geq 0,$$

and

$$f(\mathbf{0}) = \sum_{j_1=1}^{J_1} \ldots \sum_{j_n=1}^{J_n} c_{j_1 j_2 \ldots j_n} B_{j_1}(0) \ldots B_{j_n}(0) = 0,$$

$$f(\mathbf{1}) = \sum_{j_1=1}^{J_1} \ldots \sum_{j_n=1}^{J_n} c_{j_1 j_2 \ldots j_n} B_{j_1}(1) \ldots B_{j_n}(1) = 1.$$

This problem involves very sparse matrices. For solving QP problems with sparse matrices we recommend OOQP sparse solver [105], see Appendix A.5.2.

In practice it is sufficient to use linear splines with 3-5 basis functions ($J_i = 3, 4, 5$), which gives good quality approximation for 2-5 variables. For more variables the method becomes impractical because the number of spline coefficients (and hence the sizes of all matrices) grows exponentially with n. On one hand it requires a large number of data, which is usually not available, on the other hand the required computing time also becomes too large.

Example 6.1. Consider a real data set $\mathcal{D}$ of 22 input-output pairs from [287], and its approximation with a bivariate tensor-product monotone spline $J_1 = J_2 = 4$. The resulting aggregation function is plotted on Fig. 6.1.

Note 6.2. Other tensor product monotone interpolation methods [52, 62, 63] can be applied to aggregation functions, although in most cases these methods are limited to two variables. There are also alternative methods for approximation of scattered data based on triangulations [118, 257], in these methods the basis functions are determined by the data. However preservation of monotonicity becomes rather complicated, and the available methods are only suitable for bivariate case.

6.2.2 Preservation of specific properties

It is important to incorporate other problem specific information into the construction of aggregation functions. Such information may be given in terms of boundary conditions, conjunctive, disjunctive or averaging behavior, symmetry and so on. In this section we describe the method from [15, 17], which accommodates these conditions for tensor product monotone splines.

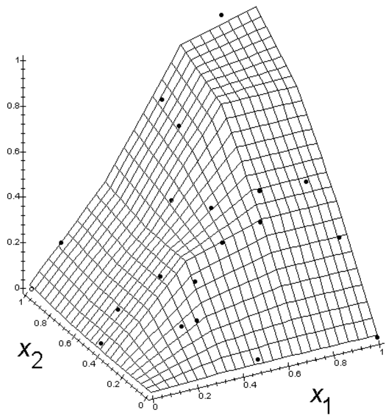


Fig. 6.1. Tensor-product spline approximation of the data from [287]. The data are marked with circles.

Symmetry

There are two methods of imposing symmetry with tensor-product splines. Consider the simplex $S = \{\mathbf{x} \in [0,1]^n | x_1 \geq x_2 \geq \ldots \geq x_n\}$ and a function $\tilde{f} : S \to [0,1]$. We recall that the function $f : [0,1]^n \to [0,1]$ defined by $f(\mathbf{x}) = \tilde{f}(\mathbf{x}_\searrow)$ is symmetric ($\mathbf{x}_\searrow$ is obtained from $\mathbf{x}$ by arranging its components in non-increasing order). Then in order to construct a symmetric f, it is sufficient to construct $\tilde{f}$.

The first method, which we call implicit, consists in constructing the tensor product splines on the whole domain, but based on the augmented data set $\tilde{\mathcal{D}}$, in which each pair $(\mathbf{x}_k, y_k)$ is replicated $n!$ times, by using the same y_k and all possible permutations of the components of $\mathbf{x}_k$. Construct monotone spline by solving QP problem (6.2). The resulting function f will be symmetric. Note that the symmetry of f is equivalent to the symmetry of the array of spline coefficients.

If the number of data becomes too large, it is possible to use the data on the simplex S only (i.e., the data set $\tilde{\mathcal{D}}$ in which each $\mathbf{x}_k$ is replaced with $\mathbf{x}_{\searrow k}$. In this case, the spline $\tilde{f}$ will approximate the data only on S, and to calculate $f(\mathbf{x})$ elsewhere one uses $f(\mathbf{x}) = \tilde{f}(\mathbf{x}_\searrow)$.

The second method is explicit, it consists in reducing the number of basis functions by the factor of $n!$ and constructing the spline $\tilde{f}$ on the simplex S using the data set $\tilde{\mathcal{D}}$ in which each $\mathbf{x}_k$ is replaced with $\mathbf{x}_{\searrow k}$. The definition of the spline (6.1) is modified to include only the products of the basis functions with the support on S. The problem (6.2) is also modified to make use of the symmetry of the array of spline coefficients.

Neutral element

We recall the definition of the neutral element (p. 12), which implies that for every $t \in [0, 1]$ in any position it holds

$$f(e, \ldots, e, t, e, \ldots, e) = t. \tag{6.3}$$

Let us use the notation $\mathbf{e}(t, i) = (e, \ldots, e, t, e, \ldots, e)$ for some $e \in [0, 1]$ with t in the i-th position.

It is shown [15] that for linear tensor product spline, it is sufficient to use the interpolation conditions $f(\mathbf{e}(t_j, i)) = t_j$ for $j = 1, \ldots, J_i$, and $i = 1, \ldots, n$, where t_j denote the knots of the spline. The conditions are imposed for all the variables. These conditions are incorporated easily into the problem (6.2) as linear equalities.

Idempotency

Condition of idempotency $f(t, \ldots, t) = t$ is equivalent to the averaging behavior of the aggregation function. For tensor-product splines this condition can be enforced by using a number of interpolation conditions $f(t_j, t_j, \ldots, t_j) = t_j, j = 1, \ldots, M$, but now t_j are not the knots of the splines. The values of t_j can be chosen with relative freedom with $M \geq n + 1$, in such a way as to match all n-dimensional rectangles (formed by tensor products of intervals between consecutive spline knots) which intersect the diagonal, see [15, 17].

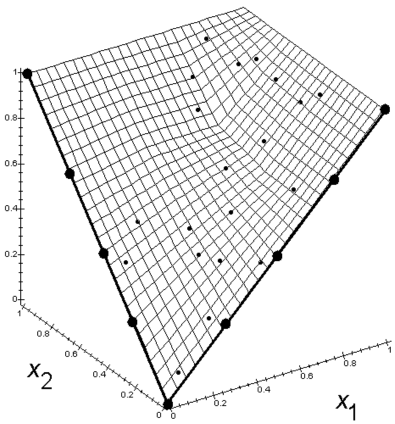


Fig. 6.2. Tensor-product spline approximation of empirical data marked with circles. The conditions of symmetry and the neutral element $e = 0$ were imposed. The data $f(0, t_j) = f(t_j, 0) = t_j, j = 1, \ldots, 5$ are marked with large filled circles.

6.3 Construction based on Lipschitz interpolation

The method of monotone Lipschitz interpolation was proposed in [21] and applied to aggregation functions in [19, 25, 27]. Denote by Mon the set of monotone non-decreasing functions on $[0,1]^n$. Then the set of general Lipschitz n-ary aggregation functions with Lipschitz constant M is characterized as

$$\mathcal{A}_{M,||\cdot||} = \{f \in Lip(M, ||\cdot||) \cap Mon : f(\mathbf{0}) = 0, f(\mathbf{1}) = 1\}.$$

We assume that the data set is consistent with the class $\mathcal{A}_{M,||\cdot||}$. If not, there are ways of smoothing the data, discussed in[21]. Our goal is to determine the *best* element of $\mathcal{A}_{M,||\cdot||}$ which interpolates the data. The *best* is understood in the sense of optimal interpolation [238]: it is the function which minimizes the worst case error, i.e., solves the following Problem

Optimal interpolation problem

$$\min_{f\in\mathcal{A}_{M,||\cdot||}} \max_{g\in\mathcal{A}_{M,||\cdot||}} \max_{\mathbf{x}\in[0,1]^n} |f(\mathbf{x}) - g(\mathbf{x})|$$

$$\text{s.t. } f(\mathbf{x}_k) = y_k, k = 1, \ldots, K.$$

The solution to this problem will be an aggregation function f which is the "center" of the set of all possible aggregation functions in this class consistent with the data.

The method of computing f is based on the following result [21].

Theorem 6.3. *Let $\mathcal{D}$ be a data set compatible with the conditions $f \in Lip(M, ||\cdot||) \cap Mon$. Then for any $\mathbf{x} \in [0,1]^n$, the values $f(\mathbf{x})$ are bounded by $\sigma_l(\mathbf{x}) \le f(\mathbf{x}) \le \sigma_u(\mathbf{x})$, with*

$$\begin{aligned} \sigma_u(\mathbf{x}) &= \min_k \{y_k + M||(\mathbf{x} - \mathbf{x}_k)_+||\}, \\ \sigma_l(\mathbf{x}) &= \max_k \{y_k - M||(\mathbf{x}_k - \mathbf{x})_+||\}, \end{aligned} \tag{6.4}$$

where $\mathbf{z}_+$ denotes the positive part of vector $\mathbf{z}$: $\mathbf{z}_+ = (\bar{z}_1, \ldots, \bar{z}_n)$, with

$$\bar{z}_i = \max\{z_i, 0\}.$$

The optimal interpolant is given by

$$f(\mathbf{x}) = \frac{1}{2}(\sigma_l(\mathbf{x}) + \sigma_u(\mathbf{x})). \tag{6.5}$$

Computation of the function f is straightforward, it requires computation of both bounds, and all the functions, σ_l, σ_u and f belong to $Lip(M, ||\cdot||) \cap Mon$. Thus, in addition to the optimal function f, one obtains as a by-product the strongest and the weakest aggregation functions from the mentioned class.

It is also useful to consider infinite data sets

$$\mathcal{D} = \{(\mathbf{t}, v(\mathbf{t})) : \mathbf{t} \in \Omega \subset [0,1]^n, v : \Omega \to [0,1]]\}$$

in which case the bounds translate into

$$B_u(\mathbf{x}) = \inf_{\mathbf{t}\in\Omega}\{v(\mathbf{t}) + M||(\mathbf{x}-\mathbf{t})_+||\},$$
$$B_l(\mathbf{x}) = \sup_{\mathbf{t}\in\Omega}\{v(\mathbf{t}) - M||(\mathbf{t}-\mathbf{x})_+||\}. \tag{6.6}$$

We will make use of these bounds when considering special properties of aggregation functions, such as idempotency or neutral element.

The function f given in Theorem 6.3 is not yet an aggregation function, because we did not take into account the conditions $f(\mathbf{0}) = 0, f(\mathbf{1}) = 1$. By adding these conditions, we obtain the following generic construction of Lipschitz aggregation functions

$$f(\mathbf{x}) = \frac{1}{2}(\underline{A}(\mathbf{x}) + \overline{A}(\mathbf{x})). \tag{6.7}$$

$$\underline{A}(\mathbf{x}) = \max\{\sigma_l(\mathbf{x}), B_l(\mathbf{x})\}, \quad \overline{A}(\mathbf{x}) = \min\{\sigma_u(\mathbf{x}), B_u(\mathbf{x})\}), \tag{6.8}$$

where the additional bounds B_l and B_u are due to specific properties of aggregation functions, considered in the next section. At the very least we have (because of $f(\mathbf{0}) = 0, f(\mathbf{1}) = 1$)

$$B_u(\mathbf{x}) = \min\{M||\mathbf{x}||, 1\}, \tag{6.9}$$
$$B_l(\mathbf{x}) = \max\{0, 1 - M||\mathbf{1} - \mathbf{x}||\},$$

but other conditions will tighten these bounds. Figure 6.3 illustrates the method of Lipschitz interpolation on the empirical data from [287].

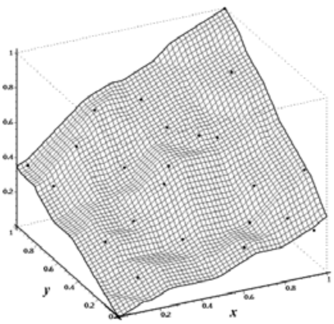


Fig. 6.3. 3D plot of an optimal Lipschitz interpolatory aggregation function based on the data from [287]. The data are marked with circles.

We note that as a special case of Equations (6.4)-(6.9) we obtain p-stable aggregation functions (Definition 1.60, p. 22), which have Lipschitz constant $M = 1$ in the norm $||\cdot||_p$. In this case the bounds (6.9) become Yager t–norm and t–conorm respectively (see p. 156), $B_u = S_p^Y, B_l = T_p^Y$.

Preservation of symmetry

Symmetry can be imposed in a straightforward manner by ordering the inputs, as discussed in Section 6.2.2. Namely, consider the simplex $S = \{\mathbf{x} \in [0,1]^n | x_1 \geq x_2 \geq \ldots \geq x_n\}$ and a function $\tilde{f} : S \to [0,1]$. The function $f : [0,1]^n \to [0,1]$ defined by $f(\mathbf{x}) = \tilde{f}(\mathbf{x}_\searrow)$ is symmetric ($\mathbf{x}_\searrow$ is obtained from $\mathbf{x}$ by arranging its components in non-increasing order). Then in order to construct a symmetric f, it is sufficient to construct $\tilde{f}$.

To build $\tilde{f}$ we simply apply Eq. (6.8), with the bounds σ_u, σ_l modified as

$$\sigma_u(\mathbf{x}) = \min_k \{y_k + M||(\mathbf{x} - \mathbf{x}_{\searrow k})_+||\},$$
$$\sigma_l(\mathbf{x}) = \max_k \{y_k - M||(\mathbf{x}_{\searrow k} - \mathbf{x})_+||\},$$

i.e., we order the abscissae of each datum in non-increasing order. There is no need to modify any of the subsequent formulae for B_u, B_l, as long as the conditions which define these bounds are consistent with the symmetry themselves (B_u, B_l will be automatically symmetric).

6.4 Preservation of specific properties

In this section we develop tight upper and lower bounds, B_l and B_u on various Lipschitz aggregation functions with specific properties. These bounds apply irrespectively of which interpolation or approximation method is used, and must be taken into account by any construction algorithm. We develop such bounds by using (6.6) with different choices of the subset Ω, corresponding to the indicated properties.

We already know that the bounds given in (6.9) apply universally to any Lipschitz aggregation function with the Lipschitz constant M. However the set of general Lipschitz aggregation functions $\mathcal{A}_{M,||\cdot||}$ is very broad, and as a consequence, the bounds in (6.9) define a wide range of possible values of an aggregation function. Often in applications there are specific properties that must be taken into account, and these properties may substantially reduce the set of functions $\mathcal{A}_{M,||\cdot||}$, and consequently produce tighter bounds. In this section we examine various generic properties, and show how the corresponding bounds B_l and B_u are computed.

6.4.1 Conjunctive, disjunctive and averaging behavior

We have the following restrictions on $[0,1]^n$, see p.9.

- Conjunctive behavior implies $f \le \min$.
- Disjunctive behavior implies $f \ge \max$.
- Averaging behavior (or idempotency) implies $\min \le f \le \max$.

These bounds immediately translate into the following functions B_u, B_l in (6.8)

- Conjunctive aggregation ($M \ge 1$) [2]

$$B_u(\mathbf{x}) = \min\{M||\mathbf{x}||, \min(\mathbf{x})\}, \quad B_l(\mathbf{x}) = \max\{0, 1 - M||\mathbf{1} - \mathbf{x}||\}.$$

- Disjunctive aggregation ($M \ge 1$)

$$B_u(\mathbf{x}) = \min\{M||\mathbf{x}||, 1\}, \quad B_l(\mathbf{x}) = \max\{1 - M||\mathbf{1} - \mathbf{x}||, \max(\mathbf{x})\}.$$

- Averaging aggregation

$$B_u(\mathbf{x}) = \min\{M||\mathbf{x}||, \max(\mathbf{x})\}, \quad B_l(\mathbf{x}) = \max\{1 - M||\mathbf{1} - \mathbf{x}||, \min(\mathbf{x})\}.$$

6.4.2 Absorbing element

The existence of an absorbing element $a \in [0, 1]$ does not imply conjunctive or disjunctive behavior on any part of the domain, but together with monotonicity, it implies $f(\mathbf{x}) = a$ on $[a, 1] \times [0, a]$ and $[0, a] \times [a, 1]$ (and their multivariate extensions).

Such restrictions are easily incorporated into the bounds by using

$$B_l(\mathbf{x}) = \max_{i=1,\dots,n} B_l^i(\mathbf{x}), \quad B_u(\mathbf{x}) = \min_{i=1,\dots,n} B_u^i(\mathbf{x}), \tag{6.10}$$
$$B_l^i(\mathbf{x}) = a - M(a - x_i)_+,$$
$$B_u^i(\mathbf{x}) = a + M(x_i - a)_+.$$

Note 6.4. Construction of aggregation functions with given *absorbent tuples*, a generalization of the absorbing element, has been developed in [28].

6.4.3 Neutral element

The existence of a neutral element is a stronger condition than conjunctive/disjunctive behavior, and consequently the bounds on the values of f are tighter. We shall see that calculation of these bounds depends on the Lipschitz constant and the norm used, and frequently requires solving an optimization problem.

We recall the definition of the neutral element $e \in [0, 1]$ (p. 12), which implies that for every $t \in [0, 1]$ in any position it holds

[2] For conjunctive and disjunctive aggregation functions M necessarily satisfies $M \ge 1$, since at the very least, for conjunctive functions $M \ge f(1, \dots, 1) - f(0, 1, \dots, 1) = 1 - 0 = 1$, and similarly for disjunctive functions.

$$f(e,\dots,e,t,e,\dots,e)=t. \tag{6.11}$$

Let us use the notation $\mathbf{e}(t,i)=(e,\dots,e,t,e,\dots,e)$ with t in the i-th position.

The bounds implied by the condition (6.11) are

$$\begin{aligned} B_u(\mathbf{x}) &= \min_{i=1,\dots,n} B_u^i(\mathbf{x}), \\ B_l(\mathbf{x}) &= \max_{i=1,\dots,n} B_l^i(\mathbf{x}), \end{aligned} \tag{6.12}$$

where for a fixed i the bounds are (from (6.6))

$$\begin{aligned} B_u^i(\mathbf{x}) &= \min_{t\in[0,1]} (t+M||(\mathbf{x}-\mathbf{e}(t,i))_+||) \\ B_l^i(\mathbf{x}) &= \max_{t\in[0,1]} (t-M||(\mathbf{e}(t,i)-\mathbf{x})_+||) \end{aligned} \tag{6.13}$$

We need to translate these bounds into practically computable values, for which we need to find the minimum/maximum with respect to t. Since any norm is a convex function of its arguments, the expression we minimize (maximize) is also convex (concave), and hence the minimum (maximum) is unique. The following proposition establishes these optima [24, 27].

Proposition 6.5. *Given $e\in[0,1]$, $\mathbf{x}\in[0,1]^n$, $i\in\{1,\dots,n\}$, $M\geq 1$, $p\geq 1$, and a norm $||\cdot||_p$, let*

$$f_{\mathbf{x},e}(t)=t+M||((x_1-e)_+,\dots,(x_{i-1}-e)_+,(x_i-t)_+,(x_{i+1}-e)_+,\dots,(x_n-e)_+)||_p$$

The minimum of $f_{\mathbf{x},e}$ is achieved at

- $t^*=0$, *if* $M=1$;
- $t^*=x_i$, *if* $p=1$ *and* $M>1$;
- $t^*=Med\left\{0, x_i-\left(\frac{c(i)}{M^{\frac{p}{p-1}}-1}\right)^{\frac{1}{p}}, x_i\right\}$ *otherwise,*

and its value is

$$\min f_{\mathbf{x},e}(t)=\begin{cases} M(c(i)+x_i^p)^{\frac{1}{p}}, & \text{if } t^*=0,\\ x_i+(M^{\frac{p}{p-1}}-1)^{\frac{p-1}{p}}c(i)^{\frac{1}{p}}, & \text{if } t^*=x_i-\left(\frac{c(i)}{M^{\frac{p}{p-1}}-1}\right)^{\frac{1}{p}},\\ x_i+Mc(i)^{\frac{1}{p}}, & \text{if } t^*=x_i, \end{cases} \tag{6.14}$$

where $c(i)=\sum_{j\neq i}(x_j-e)_+^p$.

Corollary 6.6. *The upper bound on an aggregation function with neutral element $e\in[0,1]$ and Lipschitz constant M takes the value*

$$\overline{A}(\mathbf{x})=\min_{i=1,\dots,n}\{B_u^i(\mathbf{x}),\sigma_u(\mathbf{x})\},$$

where $B_u^i(\mathbf{x})=f_{\mathbf{x},e}(t^*)$ is given by (6.14) and σ_u is given by (6.4).

Proposition 6.7. *Given* $e \in [0,1]$, $\mathbf{x} \in [0,1]^n$, $i \in \{1,\ldots,n\}$, $M \geq 1$, $p \geq 1$, *and a norm* $||\cdot||_p$ *, let*

$$g_{\mathbf{x},e}(t) = t - M||((e-x_1)_+,\ldots,(e-x_{i-1})_+,(t-x_i)_+,(e-x_{i+1})_+,\ldots,(e-x_n)_+)||_p$$

The maximum of $g_{\mathbf{x},e}(t)$ *is achieved at*

- $t^* = 1$, *if* $M = 1$;
- $t^* = x_i$, *if* $p = 1$ *and* $M > 1$, *or*
- $t^* = Med\left\{x_i, x_i + \left(\frac{\tilde{c}(i)}{M^{\frac{p}{p-1}}-1}\right)^{\frac{1}{p}}, 1\right\}$ *otherwise,*

and its value is

$$\max g_{\mathbf{x},e}(t) = \begin{cases} x_i - M\tilde{c}(i)^{\frac{1}{p}}, & \text{if } t^* = x_i, \\ x_i - (M^{\frac{p}{p-1}} - 1)^{\frac{p-1}{p}}\tilde{c}(i)^{\frac{1}{p}}, & \text{if } t^* = x_i + \left(\frac{\tilde{c}(i)}{M^{\frac{p}{p-1}}-1}\right)^{\frac{1}{p}}, \\ 1 - M(\tilde{c}(i) + (1-x_i)^p)^{\frac{1}{p}}, & \text{if } t^* = 1, \end{cases} \tag{6.15}$$

where $\tilde{c}(i) = \sum_{j \neq i}(e - x_j)_+^p$.

Corollary 6.8. *The lower bound on an aggregation function with neutral element* $e \in [0,1]$ *and Lipschitz constant* M *takes the value*

$$\underline{A}(\mathbf{x}) = \max_{i=1,\ldots,n}\{B_l^i(\mathbf{x}), \sigma_l(\mathbf{x})\}$$

where $B_l^i(\mathbf{x}) = g_{\mathbf{x},e}(t^*)$ *is given by (6.15) and* σ_l *is given by (6.4).*

Note 6.9. Recently construction of aggregation functions with given *neutral tuples*, a generalization of the neutral element, has been developed in [29].

6.4.4 Mixed behavior

In this section we develop bounds specific to the following types of mixed aggregation functions, for a given $\mathbf{e} = (e, e, \ldots, e)$.

I. f is conjunctive for $\mathbf{x} \leq \mathbf{e}$ and disjunctive for $\mathbf{x} \geq \mathbf{e}$;
II. f is disjunctive for $\mathbf{x} \leq \mathbf{e}$ and conjunctive for $\mathbf{x} \geq \mathbf{e}$;
III. f is disjunctive for $\mathbf{x} \leq \mathbf{e}$ and idempotent for $\mathbf{x} \geq \mathbf{e}$;
IV. f is conjunctive for $\mathbf{x} \leq \mathbf{e}$ and idempotent for $\mathbf{x} \geq \mathbf{e}$;
V. f is idempotent for $\mathbf{x} \leq \mathbf{e}$ and disjunctive for $\mathbf{x} \geq \mathbf{e}$;
VI. f is idempotent for $\mathbf{x} \leq \mathbf{e}$ and conjunctive for $\mathbf{x} \geq \mathbf{e}$.

We take for convenience $\mathbf{e} = (\frac{1}{2}, \ldots, \frac{1}{2})$, but we note that e is not the neutral element of f (the existence of the neutral element is not necessary). We will not require specific behavior for the values of $\mathbf{x}$ in the rest of the domain $\mathcal{R} = [0,1]^n \setminus ([0,e]^n \cup [e,1]^n)$. The reason is that in most cases the restrictions on that part of the domain will follow automatically.

We also note that cases V and VI are symmetric to cases IV and III respectively, and the results are obtained by duality.

Case I: f is conjunctive for $\mathbf{x} \le \mathbf{e}$ and disjunctive for $\mathbf{x} \ge \mathbf{e}$

In this case the behavior of the aggregation function is similar to that of a uninorm (see Fig.4.1), but no associativity is required. Yager [270] calls such a class of functions generalized uninorms (GenUNI). However our conditions are weaker, since we did not require symmetry nor the neutral element $e = 1/2$. If we did require neutral element e, the aggregation function would have the bounds given by Corollaries 6.6 and 6.8 in Section 6.4.3. Finally we note that the bounds and the optimal aggregation function are Lipschitz continuous (cf. uninorms).

On $[0, e]^n$ f is bounded from above by the minimum, and on $[e, 1]^n$ it is bounded from below by the maximum. This implies $M \ge 1$. Examine the bounds on the rest of the domain $\mathcal{R}$. Consider the lower bound. The bounds on $[0, e]^n$ imply a trivial bound $0 \le f(\mathbf{x})$ elsewhere. However, since on $[e, 1]^n$ $f(\mathbf{x}) \ge \max(\mathbf{x})$, this implies

$$f(\mathbf{x}) \ge \max_{\mathbf{z} \in [e,1]^n} (\max_i(z_i) - M||(\mathbf{z} - \mathbf{x})_+||).$$

After some technical calculations we obtain

$$\begin{aligned} f(\mathbf{x}) \ge \max_{x_k \le t \le 1} (t - M||(&\max\{0, e - x_1\}, \ldots, \\ &\max\{0, e - x_{k-1}\}, (t - x_k), \max\{0, e - x_{k+1}\}, \ldots, \max\{0, e - x_n\})||) \end{aligned} \tag{6.16}$$

for some $e \in [0, 1]$, where $x_k = \max_{i=1,\ldots,n}\{x_i\}$.

Applying Proposition 6.7, the point of maximum

- $t^* = 1$, if $M = 1$;
- $t^* = x_k$, if $p = 1$ and $M > 1$, or
- $t^* = \text{Med}\left\{x_k, x_k + \left(\frac{K}{M^{\frac{p}{p-1}} - 1}\right)^{\frac{1}{p}}, 1\right\}$ otherwise,

with $K = \sum_{i \ne k} \max\{0, e - x_i\}^p$. Thus the lower bound $B_l(\mathbf{x})$ is

$$B_l(\mathbf{x}) = \begin{cases} x_k - MK^{\frac{1}{p}}, & \text{if } t^* = x_k \\ x_k - (M^{\frac{p}{p-1}} - 1)^{\frac{p-1}{p}} K^{\frac{1}{p}}, & \text{if } t^* = x_k + \left(\frac{K}{M^{\frac{p}{p-1}} - 1}\right)^{\frac{1}{p}}, \\ 1 - M(K + (1 - x_k)^p)^{\frac{1}{p}}, & \text{if } t^* = 1. \end{cases} \tag{6.17}$$

Similarly, the fact that f is bounded from above by minimum on $[0, e]^n$ implies the following upper bound on the rest of the domain $\mathcal{R}$

$$f(\mathbf{x}) \le \min_{\mathbf{z} \in [0,e]^n} (\min_i(z_i) + M||(\mathbf{x} - \mathbf{z})_+||),$$

which translates into

$$f(\mathbf{x}) \le \min_{e \le t \le x_j} (t + M||(\max\{0, x_1 - e\}, \dots, \\ \max\{0, x_{j-1} - e\}, (x_j - t), \max\{0, x_{j+1} - e\}, \dots, \max\{0, x_n - e\})||), \quad (6.18)$$

where $x_j = \min_{i=1,\dots,n}\{x_i\}$. By applying Proposition 6.5 the minimizer is given by

- $t^* = 0$, if $M = 1$;
- $t^* = x_j$, if $p = 1$ and $M > 1$;
- $t^* = \text{Med}\left\{0, x_j - \left(\frac{K}{M^{\frac{p}{p-1}}-1}\right)^{\frac{1}{p}}, x_j\right\}$ otherwise,

and the upper bound is

$$B_u(\mathbf{x}) = \begin{cases} M(K + (x_j)^p)^{\frac{1}{p}}, & \text{if } t^* = 0, \\ x_j + (M^{\frac{p}{p-1}} - 1)^{\frac{p-1}{p}} K^{\frac{1}{p}}, & \text{if } t^* = x_j - \left(\frac{K}{M^{\frac{p}{p-1}}-1}\right)^{\frac{1}{p}}, \\ x_j + MK^{\frac{1}{p}}, & \text{if } t^* = x_j, \end{cases} \quad (6.19)$$

where $K = \sum_{i \ne j} \max\{0, x_i - e\}^p$.

Summarizing, for a mixed aggregation function with conjunctive/disjunctive behavior the bounds are

$$\begin{aligned} 0 \le f(\mathbf{x}) \le \min(\mathbf{x}), & \quad \text{if } \mathbf{x} \in [0, e]^n, \\ \max(x) \le f(\mathbf{x}) \le 1, & \quad \text{if } \mathbf{x} \in [e, 1]^n, \\ B_l(\mathbf{x}) \le f(\mathbf{x}) \le B_u(\mathbf{x}) & \quad \text{elsewhere}, \end{aligned} \quad (6.20)$$

with B_l, B_u given by (6.17) and (6.19).

Case II: f is disjunctive for $\mathbf{x} \le \mathbf{e}$ and conjunctive for $\mathbf{x} \ge \mathbf{e}$

In this case $f \ge \max$ on $[0, e]^n$ and $f \le \min$ on $[e, 1]^n$. We immediately obtain that f has the absorbing element $a = e$, i.e., $\forall \mathbf{x} \in [0, 1]^n, i \in \{1, \dots, n\}$: $f(\mathbf{x}) = e$ whenever any $x_i = e$. Such a function has a similar structure to nullnorms, but needs not be associative. It follows that $f(\mathbf{x}) = e$ for vectors whose components are not smaller or bigger than e. Thus the bounds are

$$\begin{aligned} \max(\mathbf{x}) \le f(\mathbf{x}) \le e, & \quad \text{if } \mathbf{x} \in [0, e]^n, \\ 0 \le f(\mathbf{x}) \le \min(\mathbf{x}), & \quad \text{if } \mathbf{x} \in [e, 1]^n, \\ f(\mathbf{x}) = e, & \quad \text{elsewhere}. \end{aligned} \quad (6.21)$$

Case III: f is disjunctive for $\mathbf{x} \le \mathbf{e}$ and idempotent for $\mathbf{x} \ge \mathbf{e}$

In this case, f is bounded by the maximum from below for $\mathbf{x} \le \mathbf{e}$, and is bounded by the minimum and maximum for $\mathbf{x} \ge \mathbf{e}$. This implies that e is the lower bound for all $\mathbf{x} \in \mathcal{R}$. At the same time, since f is bounded from above

by the maximum for all $\mathbf{x} \geq \mathbf{e}$, it will have the same bound for $\mathbf{x} \in \mathcal{R}$ due to monotonicity. Thus the bounds are

$$\begin{aligned} \max(\mathbf{x}) \leq f(\mathbf{x}) \leq e, \quad & \text{if } \mathbf{x} \in [0,e]^n, \\ \min(\mathbf{x}) \leq f(\mathbf{x}) \leq \max(\mathbf{x}), \quad & \text{if } \mathbf{x} \in [e,1]^n, \\ e \leq f(\mathbf{x}) \leq \max(\mathbf{x}), \quad & \text{elsewhere}. \end{aligned} \tag{6.22}$$

Case IV: f is conjunctive for $\mathbf{x} \leq \mathbf{e}$ and idempotent for $\mathbf{x} \geq \mathbf{e}$

In this case we obtain the bounds

$$\begin{aligned} 0 \leq f(\mathbf{x}) \leq \min(\mathbf{x}), \quad & \text{if } \mathbf{x} \in [0,e]^n, \\ \min(\mathbf{x}) \leq f(\mathbf{x}) \leq \max(\mathbf{x}), \quad & \text{if } \mathbf{x} \in [e,1]^n, \\ B_l(\mathbf{x}) \leq f(\mathbf{x}) \leq \tilde{B}_u(\mathbf{x}), \quad & \text{elsewhere}, \end{aligned} \tag{6.23}$$

where $\tilde{B}_u$ is given as

$$\tilde{B}_u(\mathbf{x}) = \min\{B_u(\mathbf{x}), \max(\mathbf{x})\},$$

and $B_u(\mathbf{x})$ is the expression in (6.19). B_l is given as

$$B_l(\mathbf{x}) = \max_{t \in [e,1]} (t - M||((t_1 - x_1)_+, \ldots, (t_n - x_n)_+||) = M||(\mathbf{e} - \mathbf{x})_+||.$$

6.4.5 Given diagonal and opposite diagonal

Denote by $\delta(t) = f(t, t, \ldots, t)$ the diagonal section of the n-ary aggregation function f. If $f \in \mathcal{A}_{M,||\cdot||_p}$, then $\delta \in Lip(Mn^{1/p}, ||\cdot||_p)$. Also δ is non-decreasing, and $\delta(0) = 0, \delta(1) = 1$. We denote by $\omega(t) = f(t, 1-t)$ the opposite diagonal section of a bivariate aggregation function. We note that $\omega \in Lip(M, ||\cdot||)$. We assume that the functions δ, ω are given and they have the required Lipschitz properties.

Diagonal section

From (6.6) it follows that

$$\begin{aligned} B_u(\mathbf{x}) &= \min_{t \in [0,1]} (\delta(t) + M||((x_1 - t)_+, \ldots, (x_n - t)_+)||), \\ B_l(\mathbf{x}) &= \max_{t \in [0,1]} (\delta(t) - M||((t - x_1)_+, \ldots, (t - x_n)_+)||). \end{aligned} \tag{6.24}$$

For the purposes of computing the values of $B_u(\mathbf{x}), B_l(\mathbf{x})$ we need to develop suitable algorithms to solve the optimization problems in (6.24) numerically. Since the function $\delta(t)$ is fairly arbitrary (we only require $\delta \in Lip(Mn^{1/p}, ||\cdot||_p) \cap Mon)$, the overall expression may possess a number of local minima. Calculation of the bounds require the global minimum, and thus

we need to use a global optimization technique, see Appendix A.5.5. We recommend using Pijavsky-Shubert method of deterministic global optimization.

To apply the Pijavsky-Shubert algorithm we need to estimate the Lipschitz constant of the objective function. Since $\delta \in Lip(Mn^{1/p}, ||\cdot||_p)$ and is non-decreasing, and the function

$$M||(x_1 - t)_+, \dots, (x_n - t)_+||$$

is in $Lip(Mn^{1/p}, ||\cdot||_p)$ and is non-increasing (we can prove this with the help of the identity $||\mathbf{x}||_p \le n^{1/p}||\mathbf{x}||_\infty$), the Lipschitz constant of the sum is $Mn^{1/p}$. Hence we use the Pijavsky-Shubert algorithm with this parameter.

In the special case of bivariate 1-Lipschitz functions (i.e., $n = 2, p = 1, M = 1$) we have [140]

$$\begin{aligned} B_u(\mathbf{x}) &= \max(x_1, x_2) + \min_{t\in[\alpha,\beta]}(\delta(t) - t), \\ B_l(\mathbf{x}) &= \min(x_1, x_2) + \max_{t\in[\alpha,\beta]}(\delta(t) - t), \end{aligned}$$

where $\alpha = \min(x_1, x_2)$, $\beta = \max(x_1, x_2)$.

For $p \to \infty$ a similar formula works for any dimension n. We have

$$B_u(\mathbf{x}) = \min_{t\in[0,\beta]}(\delta(t) + M(\max_i\{x_i\} - t)) = M\max_i\{x_i\} + \min_{t\in[0,\beta]}(\delta(t) - Mt),$$

and

$$B_l(\mathbf{x}) = \max_{t\in[\alpha,1]}(\delta(t) - M(t - \min_i\{x_i\})) = M\min_i\{x_i\} + \max_{t\in[\alpha,1]}(\delta(t) - Mt).$$

Opposite diagonal

We consider bivariate aggregation functions with given $\omega(t) = f(t, 1-t)$. The bounds are computed as

$$\begin{aligned} B_u(\mathbf{x}) &= \min_{t\in I}(\omega(t) + M||((x_1 - t)_+, (t - (1 - x_2))_+)||), \\ B_l(\mathbf{x}) &= \max_{t\in I}(\omega(t) - M||((t - x_1)_+, (1 - x_2 - t)_+)||). \end{aligned} \tag{6.25}$$

We notice that $\omega \in Lip(M, ||\cdot||)$ and so is the second term in the expression, hence the objective function is in $Lip(2M, ||\cdot||)$. We apply Pijavski-Shubert method with this Lipschitz parameter to calculate the values of the bounds for any $\mathbf{x}$.

As a special case the following bounds were provided for bivariate 1-Lipschitz increasing functions [140]

$$\begin{aligned} B_u(\mathbf{x}) &= T_L(\mathbf{x}) + \min_{t\in[\alpha,\beta]}(\omega(t)), \\ B_l(\mathbf{x}) &= S_L(\mathbf{x}) - 1 + \max_{t\in[\alpha,\beta]}(\omega(t)), \end{aligned} \tag{6.26}$$

where $\alpha = \min\{x_1, 1 - x_2\}$, $\beta = \max\{x_1, 1 - x_2\}$, and T_L, S_L denote Łukasiewicz t–norm and t–conorm respectively.

Example 6.10. Optimal Lipschitz aggregation functions with diagonal sections

$$\delta(t) = t^{1.4}, \quad \delta(t) = \min(1, 1.5t^2)$$

are plotted in Fig. 6.4.

Example 6.11. Optimal Lipschitz aggregation functions with opposite diagonal sections

$$\omega(t) = -t^2 + t + 0.25, \quad \omega(t) = \min(t(1-t), 0.2)$$

are plotted in Fig. 6.5.

Example 6.12. An optimal Lipschitz aggregation function with opposite diagonal section

$$\omega(t) = \min(t, (1-t))$$

is plotted in Fig. 6.6.

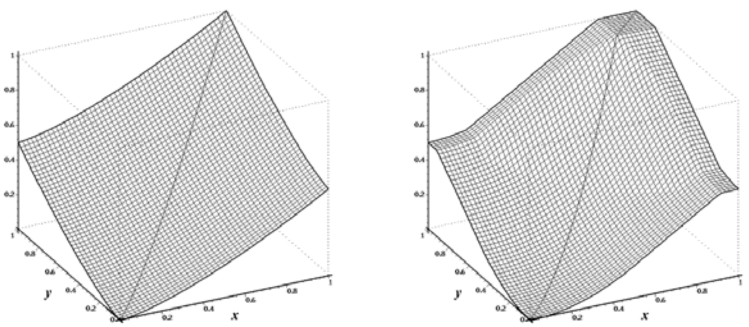


Fig. 6.4. 3D plots of optimal Lipschitz interpolatory aggregation functions with given diagonal sections in Example 6.10.

6.4.6 Given marginals

We consider the problem of obtaining an aggregation function f when certain functions are required to be its marginals. For some special cases of 1-Lipschitz aggregation functions this problem was treated in [150], the general case is presented below.

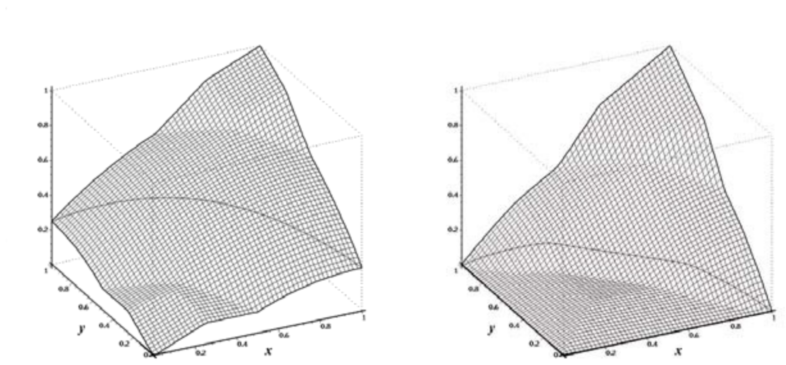


Fig. 6.5. 3D plots of optimal Lipschitz interpolatory aggregation functions with given opposite diagonal sections in Example 6.11.

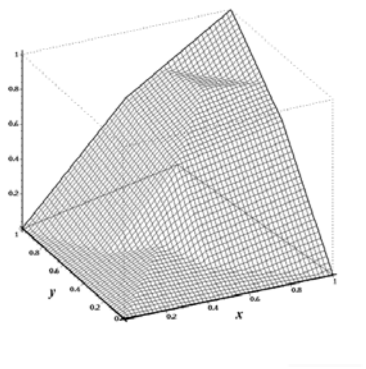


Fig. 6.6. 3D plot of an optimal Lipschitz interpolatory aggregation functions with a given opposite diagonal section in Example 6.12.

Definition 6.13. *Let* $f : [0,1]^n \to [0,1]$ *be a function. Its restriction to a subset* $\Omega = \{\mathbf{x} \in [0,1]^n | \ x_i = 0, i \in \mathcal{I}, x_j = 1; j \in \mathcal{J}, \mathcal{I} \cap \mathcal{J} = \emptyset\}$ *for some* $\mathcal{I}, \mathcal{J} \subset \{1, \ldots, n\}$ *is called a marginal function (a marginal for short)*[3].

Geometrically, the domain of a marginal is a facet of the unit cube. If $\mathcal{I} = \{1, \ldots, i-1, i+1, \ldots, n\}$ and $\mathcal{J} = \emptyset$, or vice versa, $\gamma : [0,1] \to [0,1]$ is

[3] In probability theory, a marginal probability distribution density is obtained by integrating the density $\rho(\mathbf{x})$ over all the variables x_i in $\mathcal{I} \cup \mathcal{J}$. This is equivalent to the restriction of the probability distribution function $P(\mathbf{x})$ to the set Ω in Definition 6.13 with $\mathcal{I} = \emptyset$, in the special case of the domain of P being $[0,1]^n$.

called the i-th marginal. Evidently, a marginal is a function $\gamma : [0,1]^m \to [0,1]$, $m = n - |\mathcal{I}| - |\mathcal{J}| < n$.

Consider construction of a bivariate Lipschitz aggregation function $f(x_1, x_2)$ based on a given marginal γ, defined on some closed subset Ω, for example $\Omega = \{\mathbf{x} = (x_1, x_2) | 0 \le x_1 \le 1, x_2 = 0\}$. Let $\gamma \in Lip(M_\gamma, ||\cdot||)$. Then obviously the Lipschitz constant of f, M, verifies $M \ge M_\gamma$. From (6.6) we obtain

$$\begin{aligned} B_u(\mathbf{x}) &= \min_{t\in[0,1]} (\gamma(t) + M||((x_1 - t)_+, x_2)||) \\ &= \min_{t\in[0,x_1]} (\gamma(t) + M||((x_1 - t), x_2)||), \\ B_l(\mathbf{x}) &= \max_{t\in[0,1]} (\gamma(t) - M||((t - x_1)_+, 0)||) = \gamma(x_1). \end{aligned} \tag{6.27}$$

If the marginal is given on $\Omega = \{\mathbf{x} = (x_1, x_2) | 0 \le x_1 \le 1, x_2 = 1\}$, then the bounds are

$$\begin{aligned} B_u(\mathbf{x}) &= \min_{t\in[0,1]} (\gamma(t) + M||((x_1 - t)_+, 0)||) = \gamma(x_1), \\ B_l(\mathbf{x}) &= \max_{t\in[0,1]} (\gamma(t) - M||((t - x_1)_+, 1 - x_2)||) \\ &= \max_{t\in[x_1,1]} (\gamma(t) - M||((t - x_1), 1 - x_2)||). \end{aligned} \tag{6.28}$$

To solve the optimization problem in each case we apply the Pijavski-Shubert method with the Lipschitz parameter M.

For the general multivariate case the equations are as follows. Let $\gamma_i, i = 1, \ldots, n$ be a function from $Lip(M_\gamma, ||\cdot||)$ representing the i-th marginal

$$\forall \mathbf{x} \in \Omega_i : f(\mathbf{x}) = \gamma_i(x_i), \Omega_i = \{\mathbf{x} \in [0,1]^n | x_i \in [0,1], x_j = 0, j \ne i\}.$$

The bounds resulting from the i-th marginal are

$$\begin{aligned} B_u^i(\mathbf{x}) &= \min_{t\in[0,x_i]} (\gamma_i(t) + M||(x_1, \ldots, x_{i-1}, (x_i - t)_+, x_{i+1}, \ldots, x_n)||), \\ B_l^i(\mathbf{x}) &= \gamma_i(x_i), \end{aligned}$$

and altogether we have $B_l(\mathbf{x}) = \max_{i=1,\ldots,n} B_l^i(\mathbf{x}), B_u(\mathbf{x}) = \min_{i=1,\ldots,n} B_u^i(\mathbf{x})$.

The same technique is used for construction of n-variate aggregation functions from m-variate marginals, as exemplified below. Let $\gamma : [0,1]^m \to [0,1]$ denote a marginal of f: $\forall \mathbf{x} \in \Omega : f(\mathbf{x}) = \gamma(\mathbf{y})$, with

$$\Omega = \{\mathbf{x} \in [0,1]^n | x_1, \ldots, x_m \in [0,1], x_{m+1} = \ldots = x_n = 0\}$$

and $\mathbf{y} \in [0,1]^m$, $y_i = x_i, i = 1, \ldots, m$. Then the upper and lower bounds on $f(\mathbf{x}), \mathbf{x} \in [0,1]^n \setminus \Omega$ are

$$\begin{aligned} B_u(\mathbf{x}) &= \min_{\mathbf{z}\in[0,x_1]\times\ldots\times[0,x_m]} (\gamma(\mathbf{z}) \\ &\quad + M||((x_1 - z_1)_+, \ldots, (x_m - z_m)_+, x_{m+1}, \ldots, x_n)||), \\ B_l(\mathbf{x}) &= \gamma(x_1, \ldots, x_m). \end{aligned}$$

Table 6.1. The lower and upper bounds B_l, B_u on Lipschitz aggregation functions with Lipschitz constant M and the listed properties.

Properties	$B_l(\mathbf{x})$	$B_u(\mathbf{x})$
Conjunctive function, $M \geq 1$	$\max\{1 - M\|\|\mathbf{1} - \mathbf{x}\|\|, 0\}$	$\min(\mathbf{x})$
Disjunctive function, $M \geq 1$	$\max(\mathbf{x})$	$\min\{M\|\|\mathbf{x}\|\|, 1\}$
Idempotent function	$\max\{1 - M\|\|\mathbf{1} - \mathbf{x}\|\|, \min(\mathbf{x})\}$	$\min\{M\|\|\mathbf{x}\|\|, \max(\mathbf{x})\}$
Neutral element $e \in [0,1]$, $M \geq 1$	$\max\limits_{i=1,\ldots,n}\{$ (6.15)$\}$	$\min\limits_{i=1,\ldots,n}\{$ (6.14) $\}$
Neutral element $e = 1$, $M = 1$	$\max\{1 - \|\|\mathbf{1} - \mathbf{x}\|\|, 0\}$	$\min(\mathbf{x})$
Neutral element $e = 0$, $M = 1$	$\max(\mathbf{x})$	$\min\{\|\|\mathbf{x}\|\|, 1\}$
Diagonal section $\delta(t) = f(t,t)$	given by (6.24)	
$p = 1$, $n = 2$ Diagonal section $\delta(t) = f(t,t)$	$M \min\{x_1, x_2\} + \max\limits_{t \in [\alpha,\beta]}(\delta(t) - Mt)$	$M \max\{x_1, x_2\} + \min\limits_{t \in [\alpha,\beta]}(\delta(t) - Mt)$
	$\alpha = \min\{x_1, x_2\}$, $\beta = \max\{x_1, x_2\}$	
$p \to \infty$, Diagonal section $\delta(t) = f(t,\ldots,t)$	$M \min(\mathbf{x}) + \max\limits_{t \in [\alpha,1]}(\delta(t) - Mt)$	$M \max(\mathbf{x}) + \min\limits_{t \in [0,\beta]}(\delta(t) - Mt)$
	$\alpha = \min\{x_1, x_2\}$, $\beta = \max\{x_1, x_2\}$	
Opposite diagonal $\omega(t) = f(t, 1-t)$	given by (6.25)	
$p = 1$, Opposite diagonal $\omega(t) = f(t, 1-t)$	$MS_L(\mathbf{x}) - M + \max\limits_{t \in [\alpha,\beta]}(\omega(t))$	$MT_L(\mathbf{x}) + \min\limits_{t \in [\alpha,\beta]}(\omega(t))$
	$\alpha = \min\{x_1, 1 - x_2\}$, $\beta = \max\{x_1, 1 - x_2\}$	
Marginal $g = f(\mathbf{x})\|_{\mathbf{x} \in \Omega}$, where Ω is the domain of the marginal	$\max\limits_{\mathbf{z} \in \Omega}\{g(\mathbf{z}) - M\|\|(\mathbf{z} - \mathbf{x})_+\|\|\}$	$\min\limits_{\mathbf{z} \in \Omega}\{g(\mathbf{z}) + M\|\|(\mathbf{x} - \mathbf{z})_+\|\|\}$
Mixed function, conjunctive in $[0,e]^n$, disjunctive in $[e,1]^n$	0, if $\mathbf{x} \in [0,e]^n$ $\max(\mathbf{x})$, if $\mathbf{x} \in [e,1]^n$ (6.17), elsewhere	$\min(\mathbf{x})$, if $\mathbf{x} \in [0,e]^n$ 1, if $\mathbf{x} \in [e,1]^n$ (6.19), elsewhere
Mixed function, disjunctive in $[0,e]^n$, conjunctive in $[e,1]^n$	$\max(\mathbf{x})$, if $\mathbf{x} \in [0,e]^n$ e, if $\mathbf{x} \in [e,1]^n$ e, elsewhere	e, if $\mathbf{x} \in [0,e]^n$ $\min(\mathbf{x})$, if $x \in [e,1]^n$ e, elsewhere
Mixed function, disjunctive in $[0,e]^n$, idempotent in $[e,1]^n$	$\max(\mathbf{x})$, if $\mathbf{x} \in [0,e]^n$ $\min(\mathbf{x})$, if $\mathbf{x} \in [e,1]^n$ e, elsewhere	e, if $\mathbf{x} \in [0,e]^n$ $\max(\mathbf{x})$, if $\mathbf{x} \in [e,1]^n$ $\max(\mathbf{x})$, elsewhere
Mixed function, conjunctive in $[0,e]^n$, idempotent in $[e,1]^n$	0, if $\mathbf{x} \in [0,e]^n$ $\min(\mathbf{x})$, if $\mathbf{x} \in [e,1]^n$ $M\|\|(\mathbf{e} - \mathbf{x})_+\|\|$ elsewhere	$\min(\mathbf{x})$, if $\mathbf{x} \in [0,e]^n$ $\max(\mathbf{x})$, if $\mathbf{x} \in [e,1]^n$ $\min\{\max(\mathbf{x})$,(6.19)$\}$, elsewhere

Computation of the minimum in the expression for B_u involves a nonconvex m-dimensional constrained optimization problem. There is a possibility of multiple locally optimal solutions, and the use of local descent algorithms will not deliver correct values. The proper way of calculating B_u is to use deterministic global optimization methods. We recommend using the Cutting Angle Method, see Appendix A.5.5. One should be aware that deterministic global optimization methods work reliably only in small dimension, $m < 10$. We do not expect m to be greater than 3 in applications.

6.4.7 Noble reinforcement

We recall from Section 3.7 a special class of disjunctive aggregation functions, which limit the mutual reinforcement of the inputs. We have considered several instances of the noble reinforcement requirement, namely,

1. Provide reinforcement of only *high* inputs.
2. Provide reinforcement if at least k inputs are *high*.
3. Provide reinforcement of at least k *high* inputs, if at least m of these inputs are *very high*.
4. Provide reinforcement of at least k *high* inputs, if we have *at most m low* inputs.

Of course, it is possible to combine these requirements. We have seen that an ordinal sum of t–conorm construction provides a solution to the first requirement. In this section we provide aggregation functions that satisfy the other mentioned requirements for crisp subsets of *high, very high* and *low* inputs. Fuzzification of these sets is achieved as described in Section 3.7.

Thus we concentrate on the construction of aggregation functions that satisfy conditions set in the Definitions 3.132-3.137. We will focus on Lipschitz aggregation functions with a given Lipschitz constant $M \geq 1$. We have defined three crisp thresholds α, β, γ, $\gamma < \alpha < \beta$, so that the intervals $[\alpha, 1]$, $[\beta, 1]$ and $[0, \gamma]$ denote respectively *high*, *very high* and *low* inputs.

The above mentioned requirements translate into the following aggregation functions (see pp. 191-193).

1. Reinforcement of high inputs

$$F_\alpha(\mathbf{x}) = \begin{cases} A_{i\in\mathcal{E}}(\mathbf{x}), & \text{if } \exists \mathcal{E} \subseteq \{1,\dots,n\} | \forall i \in \mathcal{E} : x_i \geq \alpha \\ & \text{and } \forall i \in \tilde{\mathcal{E}} : x_i < \alpha, \\ \max(\mathbf{x}) & \text{otherwise,} \end{cases} \tag{6.29}$$

2. Reinforcement of k high inputs

$$F_{\alpha,k}(\mathbf{x}) = \begin{cases} A_{i\in\mathcal{E}}(\mathbf{x}), & \text{if } \exists \mathcal{E} \subseteq \{1,\dots,n\} |\ |\mathcal{E}| \geq k, \\ & \qquad \forall i \in \mathcal{E} : x_i \geq \alpha, \\ & \qquad \text{and } \forall i \in \tilde{\mathcal{E}} : x_i < \alpha, \\ \max(\mathbf{x}) & \text{otherwise,} \end{cases} \tag{6.30}$$

3. Reinforcement of k high inputs with at least m very high inputs

$$F_{\alpha,\beta,k,m}(\mathbf{x}) = \begin{cases} A_{i\in\mathcal{E}}(\mathbf{x}), & \text{if } \exists\mathcal{E} \subseteq \{1,\ldots,n\} |\ |\mathcal{E}| \geq k, \\ & \forall i \in \mathcal{E} : x_i \geq \alpha, \forall i \in \tilde{\mathcal{E}} : x_i < \alpha, \\ & \text{and } \exists\mathcal{D} \subseteq \mathcal{E} |\ |\mathcal{D}| = m, \\ & \forall i \in \mathcal{D} : x_i > \beta, \\ \max(\mathbf{x}) & \text{otherwise,} \end{cases} \tag{6.31}$$

4. Reinforcement of k high inputs with no more than m low inputs

$$F_{\alpha,\gamma,k,m}(\mathbf{x}) = \begin{cases} A_{i\in\mathcal{E}}(\mathbf{x}), & \text{if } \exists\mathcal{E} \subseteq \{1,\ldots,n\} |\ |\mathcal{E}| \geq k, \\ & \forall i \in \mathcal{E} : x_i \geq \alpha, \forall i \in \tilde{\mathcal{E}} : x_i < \alpha, \\ & \text{and } \exists\mathcal{D} \subseteq \{1,\ldots,n\}| \\ & |\mathcal{D}| = n-m, \forall i \in \mathcal{D} : x_i \geq \gamma, \\ \max(\mathbf{x}) & \text{otherwise,} \end{cases} \tag{6.32}$$

where $A_{i\in\mathcal{E}}(\mathbf{x})$ is a disjunctive aggregation function, applied only to the components of $\mathbf{x}$ with the indices in $\mathcal{E}$.

Reinforcement of high inputs

Consider condition (6.29) with a fixed α. Denote by $\mathcal{E}$ a subset of indices $\{1,\ldots,n\}$ and by $\tilde{\mathcal{E}}$ its complement. For $k = 0,\ldots,n$, denote by E_k the set of points in $[0,1]^n$ which have exactly k coordinates greater than α, i.e.,

$$\begin{aligned} E_k = \{\mathbf{x} \in [0,1]^n | \exists\mathcal{E}, \text{ such that } |\mathcal{E}| = k, \\ \forall i \in \mathcal{E} : \ \alpha < x_i \text{ and } \forall j \in \tilde{\mathcal{E}} : x_j \leq \alpha\}. \end{aligned}$$

The subsets E_k form a non-intersecting partition of $[0,1]^n$. Further, $E_0 \cup E_1 \cup \ldots \cup E_k$ is a compact set.

Eq. (6.29) reads that $F_\alpha(\mathbf{x}) = \max(\mathbf{x})$ on E_0, and $F_\alpha(\mathbf{x}) \geq \max(\mathbf{x})$ on the rest of the domain, and further $F_\alpha(\mathbf{x}) \geq F_\alpha(\mathbf{y})$ for all $\mathbf{x} \in E_k, \mathbf{y} \in E_m$, $k > m$. The latter is due to monotonicity with respect to argument cardinality. Also, since no reinforcement can happen on the subset E_1, we have $F_\alpha(\mathbf{x}) = \max(\mathbf{x})$ on $E_1 \cup E_0$. This expresses the essence of the noble reinforcement requirement.

Let us now determine the upper and lower bounds B_u, B_l on F_α from (6.6). We use $\Omega = E_1 \cup E_0 \cup \{(1,\ldots,1)\}$, as this is the set on which the values of the aggregation function are specified. The datum $F_\alpha(1,\ldots,1) = 1$ implies the upper bound $F_\alpha(\mathbf{x}) \leq 1$. The general lower bound is $F_\alpha(\mathbf{x}) \geq \max(\mathbf{x})$ due to its disjunctive character. Now we need to find the upper bound B_u which results from the condition $F_\alpha = \max$ on $E_1 \cup E_0$.

Thus for any fixed $\mathbf{x}$ we need to compute

$$B_u(\mathbf{x}) = \min_{\mathbf{z}\in E_1\cup E_0} \{\max(\mathbf{z}) + M||(\mathbf{x}-\mathbf{z})_+||_p\}.$$

This is a multivariate nonlinear optimization problem, which can be reduced to n univariate problems, which are easy to solve. Consider $\mathbf{x} \in E_k$, for

a fixed k, $1 < k \leq n$, which means that k components of $\mathbf{x}$ are greater than α. Let $j \in \mathcal{E}$ be some index such that $x_j > \alpha$. It was shown in [24, 26] that the minimum is achieved at z^* whose j-th component $z_j^* \in [\alpha, x_j]$ is given by the solution to a univariate optimization problem (Eq. (6.33) below), and the rest of the components are fixed at $z_i^* = \alpha, i \neq j$. That is, we only need to find the optimal value of the component z_j, and then take minimum over all $j \in \mathcal{E}$.

Denote by $\gamma_j^* = \sum_{i \in \mathcal{E}, i \neq j} (x_i - \alpha)^p$. We have $k = |\mathcal{E}|$ univariate problems

$$B_u(\mathbf{x}) = \min_{j \in \mathcal{E}} \min_{\alpha \leq z_j \leq x_j} \{z_j + M(\gamma_j^* + (x_j - z_j)^p)^{1/p}\}. \tag{6.33}$$

For a fixed j the objective function is a convex function of z_j and hence will have a unique minimum (possibly many minimizers). A modification of Proposition 6.5 establishes this minimum explicitly

Proposition 6.14. *Let $\gamma \geq 0$, $M \geq 1$, $p \geq 1$, $\alpha, \beta \in [0, 1]$, $\alpha \leq \beta$ and*

$$f_\beta(t) = t + M((\beta - t)_+^p + \gamma)^{1/p}.$$

The minimum of $f_\beta(t)$ on $[\alpha, \beta]$ is achieved at

- $t^* = \alpha$, *if* $M = 1$;
- $t^* = \beta$, *if* $p = 1$ *and* $M > 1$;
- $t^* = Med\left\{\alpha, \beta - \left(\frac{\gamma}{M^{\frac{p}{p-1}} - 1}\right)^{\frac{1}{p}}, \beta\right\}$ *otherwise,*

and its value is

$$\min f_\beta(t) = \begin{cases} M(\gamma + (\beta - \alpha)^p)^{\frac{1}{p}}, & \text{if } t^* = \alpha, \\ \beta + M\gamma^{\frac{1}{p}}, & \text{if } t^* = \beta, \\ \beta + (M^{\frac{p}{p-1}} - 1)^{\frac{p-1}{p}} \gamma^{\frac{1}{p}} & \text{otherwise.} \end{cases} \tag{6.34}$$

Example 6.15. Consider the case of 1-Lipschitz aggregation functions $M = 1, p = 1$. Define the subset $\mathcal{E} \subseteq \{1, \ldots, n\}$ as in (6.29). The minimum in (6.33) is achieved at $z_j = \alpha$ for every $j \in \mathcal{E}$. The upper bound B_u is given as

$$B_u(\mathbf{x}) = \min(1, \alpha + \sum_{i | x_i > \alpha} (x_i - \alpha)).$$

The largest 1-Lipschitz aggregation function with noble reinforcement with threshold α is given as

$$F_\alpha(\mathbf{x}) = \begin{cases} \min\{1, \alpha + \sum\limits_{i | x_i > \alpha} (x_i - \alpha)\}, & \text{if } \exists i : x_i > \alpha, \\ \max(\mathbf{x}) & \text{otherwise,} \end{cases} \tag{6.35}$$

which is the ordinal sum of Łukasiewicz t-conorm and max.

The optimal aggregation function is

$$f(\mathbf{x}) = \frac{1}{2}(\max(\mathbf{x}), F_\alpha(\mathbf{x})),$$

which is no longer a t-conorm.

We would like to mention that even though we did not use associativity, the aggregation function F_α is defined for any number of arguments in a consistent way, preserving the existence and the value of the neutral element $e = 0$. To pass from a crisp threshold α to a fuzzy set *high*, we apply an earlier equation (3.29), with function A replaced by F_α, and α taking values in the discrete set $\{x_1, \ldots, x_n\}$.

Reinforcement of k high inputs

Now consider function (6.30) which involves cardinality of $\mathcal{E}$. It reads that $F_{\alpha,k}(\mathbf{x})$ is maximum whenever less than k components of $\mathbf{x}$ are greater or equal than α. Therefore we use the interpolation condition $F_{\alpha,k}(\mathbf{x}) = \max(\mathbf{x})$ on $\Omega = E_0 \cup E_1 \cup \ldots \cup E_{k-1}$. As earlier, B_u is given by

$$B_u(\mathbf{x}) = \min_{\mathbf{z} \in \Omega = E_0 \cup E_1 \cup \ldots \cup E_{k-1}} \{\max(\mathbf{z}) + M||(\mathbf{x} - \mathbf{z})_+||_p\}. \tag{6.36}$$

Let us compute this bound explicitly. We have an n-variate minimization problem which we intend to simplify. As earlier, $\mathbf{x}$ is fixed and $\mathcal{E}$ denotes the subset of components of $\mathbf{x}$ greater than α, $\tilde{\mathcal{E}}$ denotes its complement and $|\mathcal{E}| \geq k$. The minimum with respect to those components of $\mathbf{z}$ whose indices are in $\tilde{\mathcal{E}}$ is achieved at any $z_i^* \in [x_i, \alpha], i \in \tilde{\mathcal{E}}$. So we fix these components, say, at $z_i^* = \alpha$ and concentrate on the remaining part of $\mathbf{z}$.

At most $k-1$ of the remaining components of z are allowed to be greater than α when z ranges over Ω, we denote them by $z_{\mathcal{K}_1}, \ldots, z_{\mathcal{K}_{k-1}}, \mathcal{K} \subset \mathcal{E}, |\mathcal{K}| = k-1$. The minimum with respect to the remaining components is achieved at $z_i^* = \alpha, i \notin \mathcal{K}$. Now take all possible subsets $\mathcal{K} \subset \mathcal{E}$ and reduce the n-variate minimization problem to a number of $k-1$-variate problems with respect to $z_{\mathcal{K}_1}, \ldots, z_{\mathcal{K}_{k-1}}$

$$B_u(\mathbf{x}) = \min_{\mathcal{K} \subset \mathcal{E}, |\mathcal{K}| = k-1} \min_{z_i, i \in \mathcal{K}} \{\max_{i \in \mathcal{K}}(z_i) + M(\gamma_{\mathcal{K}}^* + \sum_{i \in \mathcal{K}} (x_i - z_i)_+^p)^{1/p}\}.$$

where $\gamma_{\mathcal{K}}^* = \sum_{i \in \mathcal{E} \setminus \mathcal{K}} (x_i - \alpha)^p$.

It was shown in [26] that the minimum for a fixed $\mathcal{K}$ is achieved when all the variables $z_i, i \in \mathcal{K}$ are equal, and hence we obtain univariate minimization problems with $t = z_{\mathcal{K}_1}$

$$B_u(\mathbf{x}) = \min_{\mathcal{K} \subset \mathcal{E}, |\mathcal{K}| = k-1} \min_{t \in [\alpha, x_{\mathcal{K}_1}]} \{t + M(\gamma_{\mathcal{K}}^* + \sum_{i \in \mathcal{K}} (x_i - t)_+^p)^{1/p}\}. \tag{6.37}$$

The minimum over all subsets $\mathcal{K}$ in (6.37) has to be computed exhaustively. The problem with respect to t is convex and piecewise smooth. It can be easily solved by the golden section method, and in some cases explicitly.

For the special case $p = 1, M \geq 1$ we have $t = x_{\mathcal{K}_1}$ and

$$B_u(\mathbf{x}) = \min_{\mathcal{K} \subset \mathcal{E}, |\mathcal{K}|=k-1} (x_{\mathcal{K}_1} + M \sum_{i \in \mathcal{E} \setminus \mathcal{K}} (x_i - \alpha)). \tag{6.38}$$

For $p = 2$ we obtain a quadratic equation in t which we also can solve explicitly.

Example 6.16. Consider again the case of 1-Lipschitz aggregation functions $M = 1, p = 1$, for $n = 4$ and the requirement that at least $k = 3$ high arguments reinforce each other. We reorder the inputs as $x_{(1)} \geq x_{(2)} \geq x_{(3)} \geq x_{(4)}$. Applying (6.38) we have the strongest 1-Lipschitz aggregation function

$$F_\alpha(\mathbf{x}) = \begin{cases} \min\{1, x_{(1)} + x_{(3)} - \alpha\}, & \text{if } x_{(4)} \leq \alpha \text{ and} \\ & x_{(1)}, x_{(2)}, x_{(3)} > \alpha, \\ \min\{1, x_{(1)} + x_{(3)} + x_{(4)} - 2\alpha\}, & \text{if all } x_{(1)}, \ldots, x_{(4)} > \alpha, \\ \max(\mathbf{x}) & \text{otherwise.} \end{cases} \tag{6.39}$$

Note the absence of $x_{(2)}$ (the subsets $\mathcal{E} \setminus \mathcal{K}$ in (6.38) have cardinality 2 when $x_{(4)} > \alpha$ and 1 when $x_{(4)} \leq \alpha$). The optimal aggregation function is

$$f(\mathbf{x}) = \frac{1}{2}(\max(\mathbf{x}), F_\alpha(\mathbf{x})).$$

The algorithmic implementation of (6.30) or (6.37), see Figure 6.7, is straightforward (the former is a special case of the latter).

Note 6.17. When the noble reinforcement property does not involve the minimum cardinality k of the set of *high* arguments (property (6.29)), use $k = 2$.

Reinforcement of k high inputs with at least m very high inputs

We consider calculation of the bound B_u in (6.31). We proceed as earlier, but note that the definition of the subset Ω where the value of $F_{\alpha,\beta,k,m}$ is restricted to *max* has changed. Fortunately, we can still use the algorithms from the previous section as follows. We can write $\Omega = \Omega_1 \cup \Omega_2$, where $\Omega_1 = E_0 \cup E_1 \cup \ldots, \cup E_{k-1}$ as in (6.37), and

$$\begin{aligned} \Omega_2 = \{\mathbf{x} \in [0,1]^n | \ \exists \mathcal{D} \text{ such that} \\ |\mathcal{D}| = m, \forall i \in \{1, \ldots, n\} \setminus \mathcal{D} : x_i \leq \beta\}, \end{aligned}$$

i.e., Ω_2 is the set of points that have less than m coordinates greater than β. According to (6.31), $F_{\alpha,\beta,k,m}$ is restricted to max on that subset.

Next we show that the bound B_u can be easily computed by adapting our previous results. First, note that

Algorithm 1
Purpose: Find the upper bound $B_u(\mathbf{x})$ given by (6.36).
Input: Vector $\mathbf{x}$, threshold α, subset $\mathcal{E}$, cardinality k, Lipschitz constant M, norm $||\cdot||_p$.
Output: The value $B_u(\mathbf{x})$.

Step 1 For every subset $\mathcal{K} \in \mathcal{E}, |\mathcal{K}| = k-1$ do
Step 1.1 Compute $\gamma^*_{\mathcal{K}} = \sum\limits_{i\in\mathcal{E}\setminus\mathcal{K}} (x_i - \alpha)^p$.
Step 1.2 Compute the largest component $x_{\mathcal{K}_1} = \max_{i\in\mathcal{K}} x_i$.
Step 1.3 Find the minimum $\sigma_{\mathcal{K}} = \min\limits_{t\in[\alpha, x_{\mathcal{K}_1}]} \{t + M(\gamma^*_{\mathcal{K}} + \sum\limits_{i\in\mathcal{K}} (x_i - t)^p_+)^{1/p}$
by using golden section method.
Step 2 Compute $B_u = \min\limits_{\mathcal{K}\subset\mathcal{E}} \sigma_{\mathcal{K}}$.
Step 3 Return B_u.

Algorithm 2
Purpose: Compute the value of an aggregation function with noble reinforcement of at least k components (6.30).
Input: Vector $\mathbf{x}$, threshold α, the minimum cardinality k of the subset of reinforcing arguments, Lipschitz constant M, norm $||\cdot||_p$.
Output: The value $F_{\alpha,k}(\mathbf{x})$.

Step 1 Compute the subset of indices $\mathcal{E} = \{i | x_i > \alpha\}$.
Step 2 Call Algorithm $1(\mathbf{x}, \alpha, E, k, M, p)$ and save the output in B_u.
Step 3 Compute $F_{\alpha,k} = \frac{\min\{1, B_u\} + \max(\mathbf{x})}{2}$.
Step 4 Return $F_{\alpha,k}$.

Fig. 6.7. Algorithmic implementation of Eqs. (6.30) and (6.37)

$$
\begin{aligned}
B_u(\mathbf{x}) &= \min_{\mathbf{z}\in\Omega}\{\max(\mathbf{z}) + M||(\mathbf{x}-\mathbf{z})_+||_p\} \\
&= \min\left\{\min_{\mathbf{z}\in\Omega_1}\{\max(\mathbf{z}) + M||(\mathbf{x}-\mathbf{z})_+||_p\}, \min_{\mathbf{z}\in\Omega_2}\{\max(\mathbf{z}) + M||(\mathbf{x}-\mathbf{z})_+||_p\}\right\}.
\end{aligned}
$$

We already know how to compute the minimum over Ω_1 by using (6.37).

Consider a partition of $[0,1]^n$ into subsets

$$
\begin{aligned}
D_j = \{\mathbf{x} \in [0,1]^n | \ &\exists \mathcal{D} \text{ such that } |\mathcal{D}| = j, \\
&\forall i \in \mathcal{D}, \beta < x_i \le 1, \forall j \in \tilde{\mathcal{D}}, x_j \le \beta\},
\end{aligned}
$$

for $j = 0, \ldots, n$. It is analogous to the partition given by E_k on p.291, with β replacing α. D_j is the set of input vectors with j *very high* scores.

Now $\Omega_2 = D_0 \cup \ldots \cup D_{m-1}$. Thus computation of the minimum over Ω_2 is completely analogous to the minimum over Ω_1 (cf. (6.36)), the only difference is that we take $m < k$ instead of k and β rather than α. Hence we apply the solution given in (6.37) for $m > 1$.

The special case $m = 1$, i.e., the requirement "at least one score should be *very high*" is treated differently. In this case $\Omega_2 = D_0$ and solution (6.37) is not

applicable. But in this case an optimal solution is [24, 26] $\mathbf{z}^* = (\beta, \beta, \ldots, \beta)$. The value of the minimum in this case is

$$\min_{\mathbf{z}\in D_0} \{\max(\mathbf{z}) + M||(\mathbf{x}-\mathbf{z})_+||_p\} = \beta + M(\sum_{i|x_i>\beta} (x_i - \beta)^p)^{1/p}.$$

Reinforcement of k high inputs with at most m low inputs

Here we have $1 < k \le n$ and $0 \le m \le n-k$; when $m = 0$ we prohibit reinforcement when at least one low input is present. We consider calculation of the bound B_u in (6.32).

We proceed similarly to the previous case. Form a partition of $[0,1]^n$: for $j = 0, \ldots n$ define

$$D_j = \{\mathbf{x} \in [0,1]^n | \ \exists \mathcal{D} \subseteq \{1, \ldots, n\} \text{ such that } |\mathcal{D}| = j, \\ \forall i \in \mathcal{D}, \ \gamma < x_i \le 1, \forall j \in \tilde{\mathcal{D}}, x_j \le \gamma\}.$$

D_j is the set of points with $n-j$ *small* coordinates, and the aggregation function $F_{\alpha,\gamma,k,m}$ should be restricted to maximum on $\Omega_3 = D_0 \cup \ldots \cup D_{n-m-1}$, as well as on $\Omega_1 = E_0 \cup \ldots \cup E_{k-1}$, hence

$$\begin{aligned} B_u(\mathbf{x}) &= \min_{\mathbf{z}\in\Omega}\{\max(\mathbf{z}) + M||(\mathbf{x}-\mathbf{z})_+||_p\} \\ &= \min\left\{\min_{\mathbf{z}\in\Omega_1}\{\max(\mathbf{z}) + M||(\mathbf{x}-\mathbf{z})_+||_p\}, \min_{\mathbf{z}\in\Omega_3}\{\max(\mathbf{z}) + M||(\mathbf{x}-\mathbf{z})_+||_p\}\right\}. \end{aligned}$$

where the minimum over Ω_1 is computed by using (6.37), and the minimum over Ω_3 is computed analogously (by replacing α with γ and k with $n-m$).

All the requirements discussed in this section led to defining the subset Ω on which the aggregation function coincides with the maximum. In all cases we used the basic equations (6.4), and reduced the n-variate minimization problems to a number of univariate problems, of the same type as (6.37), with different parameters. Numerical computation of the bounds B_u is done efficiently by applying algorithms on Fig. 6.7 with different parameters.

7

Other Types of Aggregation and Additional Properties

In this concluding chapter we will give a brief overview of several types of aggregation functions which have not been mentioned so far, as well as pointers to selected literature. There are many specific types of aggregation functions developed in the recent years, and it is outside the scope of this book to have their exhaustive overview. We also briefly summarize some mathematical properties of aggregation functions not mentioned in Chapter 1. Some of the recent developments are covered in two new monographs on the topic [114, 237].

7.1 Other types of aggregation

Aggregation functions with flying parameter

We have studied several families of aggregation functions which depend on some parameter. For instance power means (Definition 2.12), families of t–norms and t–conorms (Section 3.4.11), families of copulas (Section 3.5), families of uninorms (Section 4.2.3), T-S functions (Section 4.5) and some others. In all cases the parameter which characterizes a specific member of each family did not depend on the input. An obvious question arises: what if the parameter is also a function of the inputs? This leads to aggregation functions with flying parameter [43], p.43, [275].

Proposition 7.1. *Let f_r, $r \in [0,1]$ be a family of aggregation functions, such that $f_{r_1} \leq f_{r_2}$ as long as $r_1 \leq r_2$, and let g be another aggregation function. Then the function*

$$f_g(x_1, \ldots, x_n) = f_{g(x_1,\ldots,x_n)}(x_1, \ldots, x_n)$$

is also an aggregation function. Further, if f_r is a family of conjunctive, disjunctive, averaging or mixed aggregation functions, f_g also belongs to the respective class.

The same result is valid for extended aggregation functions. Note that many families of aggregation functions are defined with respect to a parameter ranging over $[0,\infty]$ or $[-\infty,\infty]$. To apply flying parameter construction, it is possible to redefine parameterizations using some strictly increasing bijection $\varphi : [0,1] \to [0,\infty]$ (or $\to [-\infty,\infty]$), for instance $\varphi(t) = \frac{t}{t+1}$. In this case we have $f_r = f_{\varphi(g(x_1,\ldots,x_n))}$.

Example 7.2. Let f_r be the Hamacher family of t–norms T^H_λ, $\lambda \in [0,\infty]$ (p. 152), and let g be the arithmetic mean M. The function

$$T^H_M(\mathbf{x}) = T^H_{\frac{M(\mathbf{x})}{M(\mathbf{x})+1}}(\mathbf{x})$$

is a conjunctive aggregation function (but it is not a t–norm).

Example 7.3. Let f_r be the family of linear convex T-S function $L_{\gamma,T,S}$ (p. 231) and let $g = T_P$, the product t–norm. The function

$$L_{T_P,T,S}(\mathbf{x}) = L_{T_P(\mathbf{x}),T,S}(\mathbf{x}) = (1 - T_P(\mathbf{x})) \cdot T(\mathbf{x}) + T_P(\mathbf{x}) \cdot S(\mathbf{x})$$

is an aggregation function. Interestingly, in the case of $T = T_P, S = S_P$, (product and dual product) and $g = U$, the $3 - \Pi$ uninorm (p. 209), we obtain [43], p.43, [275]

$$L_{U,T_P,S_P} = U.$$

Weighting vectors of various aggregation functions, such as weighted quasi-arithmetic means, OWA, weighted t–norms and t–conorms, weighted uninorms, etc., can also be made dependent on the input vector $\mathbf{x}$. Examples of such functions are the Bajraktarevic mean (Definition 2.48 on p. 66), counter-harmonic means (see p. 63), mixture functions (see p. 67) and neat OWA (Definition 2.55 on p. 73). One difficulty when making the weights dependent on $\mathbf{x}$ is that the resulting function f is not necessarily monotone non-decreasing, and hence is not an aggregation function. For mixture functions a sufficient condition for monotonicity of f was established recently in [169], see discussion on p. 67.

Averaging functions based on penalty functions

Given a vector $\mathbf{x}$, it is possible to formulate the following problem: what would be the value y which is in some sense closest to all values $x_i, i = 1,\ldots,n$. The measure of closeness is evaluated by using

$$P(\mathbf{x}, y) = \sum_{i=1}^{n} w_i p(x_i, y), \tag{7.1}$$

where $p : [0,1]^2 \to [0,\infty]$ is some "penalty", or dissimilarity, function, with the properties i) $p(t,s) = 0$ if and only if $t = s$, and ii) $p(t_1,s) \geq p(t_2,s)$

whenever $t_1 \geq t_2 \geq s$ or $t_1 \leq t_2 \leq s$, and $\mathbf{w}$ is a vector of non-negative weights [49, 183]. The value of the aggregation function f is the value y^* which minimizes the total penalty $P(\mathbf{x}, y)$ with respect to y for a given $\mathbf{x}$, $f(\mathbf{x}) = y^* = argmin_y P(\mathbf{x}, y)$. f is necessarily an averaging aggregation function.

To ensure y^* is unique, the authors of [49] use so called "faithful" penalty function, $p(t, s) = K(h(t), h(s))$, where $h : [0, 1] \to [-\infty, \infty]$ is some continuous monotone function and $K : [-\infty, \infty] \to [0, \infty]$ is convex. Under these conditions P has a unique minimum, but possibly many minimizers. y^* is then the midpoint of the set of minimizers of (7.1).

In the special case $p(t, s) = (t - s)^2$, f becomes a weighted arithmetic mean, and in the case $p(t, s) = |t - s|$ it becomes a weighted median. If $p(t, s) = (h(t) - h(s))^2$ one obtains a weighted quasi-arithmetic mean with the generator h. Many other aggregation functions, including OWA and ordered weighted medians, can also be obtained, see [183].

Note that closed form solutions to the penalty minimization problem, such as those mentioned above, are rare. In general one has to solve the optimization problem numerically. On the other hand, this method offers a great flexibility in dealing with means.

A related class of averaging functions is called *deviation means* [40], p. 316. The role of penalty functions is played by the deviation functions $d : [0, 1]^2 \to \Re$ which are continuous and strictly increasing with respect to the second argument and satisfy $d(t, t) = 0$. The equation

$$\sum_{i=1}^{n} w_i d(x_i, y) = 0$$

has a unique solution, which is the value of $f(\mathbf{x})$. If $d(t, s) = h(s) - h(t)$ for some continuous strictly monotone function h, one recovers the class of weighted quasi-arithmetic means with the generator h.

Bi-capacities

Recently the concept of a fuzzy measure was extended to set functions on a product $2^{\mathcal{N}} \times 2^{\mathcal{N}}$, $\mathcal{N} = \{1, \ldots, n\}$, called bi-capacities [112, 113]. Formally, let

$$Q(\mathcal{N}) = \{(\mathcal{A}, \mathcal{B}) \in 2^{\mathcal{N}} \times 2^{\mathcal{N}} | \mathcal{A} \cap \mathcal{B} = \emptyset\}.$$

A discrete bi-capacity is the mapping $v : Q(\mathcal{N}) \to [-1, 1]$, non-decreasing with respect to set inclusion in the first argument and non-increasing in the second, and satisfying:

$$v(\emptyset, \emptyset) = 0, v(\mathcal{N}, \emptyset) = 1, v(\emptyset, \mathcal{N}) = -1.$$

Bi-capacities are useful for aggregation on bipolar scales (on the interval $[-1, 1]$), and are used to define a generalization of the Choquet integral as an aggregation function. Bi-capacities are represented by 3^n coefficients. The Möbius transformation, interaction indices and other quantities have been defined for bi-capacities as well [112, 144, 146, 258], see also [214, 215].

Aggregation functions on sets other than intervals

In this book we were interested exclusively in aggregation functions on $[0,1]$ (or any closed interval $[a,b]$, see p. 31). However there are many constructions in which the aggregation functions are defined on other sets. We mention just two examples.

Aggregation functions can be defined on discrete sets, such as $\mathcal{S} = \{0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\}$, i.e., as functions $f : \mathcal{S}^n \to \mathcal{S}$. An example of such aggregation functions are discrete t–norms and copulas [149, 175, 176]. Another example is the class of weighted ordinal means [148].

Aggregation functions can also be defined on sets of intervals, i.e., when the arguments and the value of an aggregation function f are not numbers from $[0,1]$ but intervals. Formally, the lattice $\mathcal{L}^I = (L^I, \leq_{L^I})$ where

$$L^I = \{[a,b] | (a,b) \in [0,1]^2 \text{ and } a \leq b\},$$

$$[a,b] \leq_{L^I} [c,d] \Leftrightarrow (a \leq b \text{ and } c \leq d) \text{ for all } [a,b],[c,d] \in L^I.$$

Such aggregation functions are useful when aggregating membership values of interval-valued and intuitionistic fuzzy sets [8].

Some recent results related to interval-based t–norms can be found in [74, 75, 76].

Linguistic aggregation functions

In [126] the authors proposed the linguistic OWA function, based on the definition of a convex combination of linguistic variables [69]. They subsequently extended this approach to linguistic OWG functions [1][125], see also [123, 124, 127].

Linguistic variables are variables, whose values are labels of fuzzy sets [281, 282, 283]. The arguments of an aggregation function, such as linguistic OWA or linguistic OWG, are the labels from a totally ordered universal set $\mathcal{S}$, such as $\mathcal{S}$ ={*Very low, Low, Medium, High, Very high*}. For example, such arguments could be the assessments of various alternatives by several experts, which need to be aggregated, e.g., $f(L, M, H, H, VL)$. The result of such aggregation is a label from $\mathcal{S}$.

Multistage aggregation

Double aggregation functions were introduced in [50] with the purpose to model multistage aggregation process. They are defined as

$$f(\mathbf{x}) = a(g(\mathbf{y}), h(\mathbf{z})),$$

[1] The standard OWA and OWG functions were defined on pp. 68,73.

where a, g, h are aggregation functions, and $\mathbf{y} \in [0,1]^k$,$\mathbf{z} \in [0,1]^m$, $k + m = n$ and $\mathbf{x} = \mathbf{y}|\mathbf{z}$. "$\cdot|\cdot$" denotes concatenation of two vectors. A typical application of such operators is when the information contained in $\mathbf{y}$ and $\mathbf{z}$ is of different nature, and is aggregated in different ways. a may have more than two arguments.

Double aggregation functions can be used to model the following logical constructions *If* (A **AND** B **AND** C) **OR** (D **AND** E) *then ...*

In this process some input values are combined using one aggregation function, other arguments are combined using a different function, and at the second stage the outcomes are combined with a third function. While the resulting function f is an ordinary aggregation function, it has certain structure due to the properties of functions a, g and h, such as right- and left-symmetry, and so forth [50].

Multi-step Choquet integrals have been treated in [186, 194, 237].

7.2 Some additional properties

As we already mentioned, there are several criteria that may help in the selection of the most suitable aggregation function, and one of these criteria is the fulfillment of some specific mathematical properties. The most important properties, such as idempotency, symmetry, associativity, existence of a neutral or an absorbing element, etc., have been studied in detail in Chapter 1. In this section we briefly summarize some additional properties of aggregation functions.

Invariantness and Comparison Meaningfulness properties

The properties of homogeneity, shift-invariance, linearity and self-duality, presented in Chapter 1, are special cases of a more general property, *invariantness*, which characterizes an important class of scale-independent functions:

Definition 7.4 (Invariant aggregation function). *Given a monotone bijection $\varphi : [0,1] \to [0,1]$, an aggregation function f is called* φ-invariant *if it verifies $f = f_\varphi$, where f_φ is defined as*

$$f_\varphi(x_1, \ldots, x_n) = \varphi^{-1}(f(\varphi(x_1), \ldots, \varphi(x_n))).$$

An aggregation function is said to be invariant *when it is φ-invariant for every monotone bijection φ.*

When φ is a strong negation, φ-invariant aggregation functions are nothing but N-self-dual aggregation functions (see, e.g. [160, 161]), i.e., N-symmetric sums, which have been studied in detail in Chapter 4. On the other hand, the class of functions invariant under increasing bijections has been completely characterized in [200] (continuous case) and in [185] (general case). Note that the only invariant aggregation functions are projections.

A related property is called *comparison meaningfulness.*

Definition 7.5 (Comparison meaningful aggregation function). *Given a strictly increasing bijection* $\varphi : [0,1] \to [0,1]$*, an aggregation function* f *is called* φ-comparison meaningful *if for any* $\mathbf{x}, \mathbf{y} \in [0,1]^n$

$$f(\mathbf{x}) \quad \begin{Bmatrix} < \\ = \end{Bmatrix} \quad f(\mathbf{y}) \qquad \textit{implies} \qquad f(\varphi(\mathbf{x})) \quad \begin{Bmatrix} < \\ = \end{Bmatrix} \quad f(\varphi(\mathbf{y})),$$

where for any $\mathbf{z} \in [0,1]^n$*,* $\varphi(\mathbf{z})$ *denotes the vector* $(\varphi(z_1), \ldots, \varphi(z_n))$*. An aggregation function is said to be* comparison meaningful *when it is* φ*-comparison meaningful for every strictly increasing bijection* φ*.*

Comparison meaningful functions are presented in [168].

The Non-Contradiction and Excluded-Middle Laws

There are two well-known logical properties, the *Non-Contradiction* (NC) and the *Excluded-Middle* (EM) laws, which were studied in the context of aggregation functions from two different points of view. Focussing on Non-Contradiction, this law can be stated in its ancient Aristotelian formulation as follows: for any statement p, the statements p and *not* p cannot hold at the same time, i.e., $p \wedge \neg p$ is *impossible*, where the binary operation $\wedge$ represents the *and* connective and the unary operation $\neg$ stands for negation. This formulation can be interpreted in at least two different ways, depending on how the term *impossible* is understood [240]:

1. Taking an approach common in modern logic, the term *impossible* can be thought of as *false*, and then the NC principle can be expressed in a logical structure with the minimum element $\mathbf{0}$ as $p \wedge \neg p = \mathbf{0}$.
2. Another possibility, which is closer to ancient logic, is to interpret *impossible* as *self-contradictory* (understanding that an object is self-contradictory whenever it entails its negation). In this case, the NC principle can be written as $p \wedge \neg p \vDash \neg(p \wedge \neg p)$, where $\vDash$ represents an entailment relation.

In the context of aggregation functions, if the operation $\wedge$ is represented by means of a bivariate aggregation function $f : [0,1]^2 \to [0,1]$, and the logical negation is modeled by a strong negation[2] N, the NC law can be interpreted in the following ways:

Definition 7.6. *Let* $f : [0,1]^2 \to [0,1]$ *be a bivariate aggregation function and let* $N : [0,1] \to [0,1]$ *be a strong negation.*

- f *satisfies the NC law in modern logic w.r.t. to* N *if* $f(t, N(t)) = 0$ *holds for all* $t \in [0,1]$*.*

[2] See Definition 1.48 in Chapter 1.

- *f satisfies the NC law in ancient logic w.r.t. to N if $f(t, N(t)) \leq N(f(t, N(t)))$ holds for all $t \in [0,1]$.*

Similar arguments can be applied to the Excluded-Middle law, and they result in the following definition, dual to the Definition 7.6.

Definition 7.7. *Let $f : [0,1]^2 \to [0,1]$ be a bivariate aggregation function and let $N : [0,1] \to [0,1]$ be a strong negation.*

- *f satisfies the EM law in modern logic w.r.t. to N if $f(t, N(t)) = 1$ holds for all $t \in [0,1]$.*
- *f satisfies the EM law in ancient logic w.r.t. to N if $N(f(t, N(t))) \leq f(t, N(t))$ holds for all $t \in [0,1]$.*

The satisfaction of the NC and EM laws by various aggregation functions was studied in [208] (ancient logic interpretation) and in [207] (modern logic interpretation).

Local internality property

An aggregation function is said to be *locally internal* when it always provides as the output the value of one of its arguments:

Definition 7.8. *An aggregation function f is called* locally internal *if for all $x_1, \ldots, x_n \in [0,1]^n$, $f(x_1, \ldots, x_n) \in \{x_1, \ldots, x_n\}$.*

Evidently, any locally internal aggregation function is idempotent (and then, see Note 1.12, it is an averaging function), but not vice-versa. Projections and order statistics (see Chapter 2), along with the minimum and maximum, are trivial instances of locally internal aggregation functions. Other functions in this class are the left- and right-continuous idempotent uninorms characterized in [64]. For details on this topic, we refer the reader to [170], where bivariate locally internal aggregation functions have been characterized and studied in detail in conjunction with additional properties, such as symmetry, associativity and existence of a neutral element.

Self-identity property

In [279] Yager introduced the so-called *self-identity* property, applicable to extended aggregation functions, and defined as follows.

Definition 7.9. *An extended aggregation function F has the* self-identity property *if for all $n \geq 1$ and for all $x_1, \ldots, x_n \in [0,1]^n$,*

$$F(x_1, \ldots, x_n, F(x_1, \ldots, x_n)) = F(x_1, \ldots, x_n).$$

Note that extended aggregation functions that satisfy the self-identity property are necessarily idempotent (and hence averaging), but the converse is not true. The arithmetic mean, the a-medians (see Chapter 2) and the functions min and max are examples of extended idempotent functions with the self-identity property. A subclass of weighted means that possess this property was characterized in [279].

Extended aggregation functions with the self-identity property verify, in addition, the following two inequalities:

$$\begin{aligned} F(x_1,\dots,x_n,k) &\geq F(x_1,\dots,x_n) \quad \text{if} \quad k \geq F(x_1,\dots,x_n), \\ F(x_1,\dots,x_n,k) &\leq F(x_1,\dots,x_n) \quad \text{if} \quad k \leq F(x_1,\dots,x_n). \end{aligned}$$

A

Tools for Approximation and Optimization

In this appendix we outline some of the methods of numerical analysis, which are used as tools for construction of aggregation functions. Most of the material in this section can be found in standard numerical analysis textbooks [41, 57]. A few recently developed methods will also be presented, and we will provide the references to the articles which discuss these methods in detail.

A.1 Univariate interpolation

Consider a set of data $(x_k, y_k), k = 1, \ldots, K$, $x_k, y_k \in \Re$. The aim of interpolation is to define a function f, which can be used to calculate the values at x distinct from x_k. The interpolation conditions are specified as $f(x_k) = y_k$ for all $k = 1, \ldots, K$. We assume that the abscissae are ordered $x_k < x_{k+1}, k = 1, \ldots, K-1$.

Polynomial interpolation is the best known method. It consists in defining f as $(K-1)$-st degree polynomial

$$f(x) = a_{K-1}x^{K-1} + a_{K-2}x^{K-2} + \ldots + a_1 x + a_0,$$

and then using K interpolation conditions to determine the unknown coefficients $a_{K-1}, \ldots, a_0$. They are found by solving a linear system of equations

$$\begin{bmatrix} x_1^{K-1} & x_1^{K-2} & \ldots & x_1 & 1 \\ x_2^{K-1} & x_2^{K-2} & \ldots & x_2 & 1 \\ \vdots & & & & \vdots \\ x_K^{K-1} & x_K^{K-2} & \ldots & x_K & 1 \end{bmatrix} \begin{bmatrix} a_{K-1} \\ a_{K-2} \\ \vdots \\ a_0 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_K \end{bmatrix}.$$

It is well known that such a system of equations is ill-conditioned, and that writing down a polynomial f in Lagrange or Newton basis,

$$f(x) = b_{K-1}N_{K-1}(x) + b_{K-2}N_{K-2}(x) + \ldots + b_1 N_1(x) + b_0 N_0(x),$$

produces a different representation of the same polynomial, but with a better conditioned system of equations, a lower triangular for the Newton's basis and the identity matrix in the Lagrange basis.

It is not difficult to generalize this method for interpolation in any finite-dimensional function space. Let $B_1, B_2, \ldots, B_K$ be some linearly independent functions. They span a linear vector space V, such that any $f \in V$ can be written as

$$f(x) = \sum_{k=1}^{K} a_k B_k(x).$$

The interpolation conditions yield the linear system of equations $\mathbf{Ba} = \mathbf{y}$ with the matrix

$$\mathbf{B} = \begin{bmatrix} B_1(x_1) & B_2(x_1) & \ldots & B_K(x_1) \\ B_1(x_2) & B_2(x_2) & \ldots & B_K(x_2) \\ \vdots & & & \vdots \\ B_1(x_K) & B_2(x_K) & \ldots & B_K(x_K) \end{bmatrix}$$

The usual choices for the functions B_k are the Newton polynomials, trigonometric functions, radial basis functions and B-splines. B-splines are especially popular, because they form a basis in the space of polynomial splines.

Polynomial splines are just piecewise polynomials (i.e., on any interval $[x_k, x_{k+1}]$ f is a polynomial of degree at most n), although typically continuity of the function f itself and some of its derivatives is required. For a spline of an odd degree $n = 2m - 1$, the first $m - 1$ derivatives are continuous and the m-th derivative is square integrable. If less than m derivatives are square integrable, the spline is said to have deficiency greater than 1. Typically linear ($n = 1$) and cubic ($n = 3$) splines are used [66, 77].

The most important property of polynomial interpolating splines (of an odd degree $2m - 1$ and deficiency 1)[1] is that they minimize the following functional, interpreted as the energy, or smoothness

$$F(f) = \int_{x_1}^{x_K} |f^{(m)}(t)|^2 dt.$$

Thus they are considered as the most "smooth" functions that interpolate given data. Compared to polynomial interpolation, they do not exhibit unwanted oscillations of f for a large number of data.

B-splines are defined recursively as

$$B_k^1(x) = \begin{cases} 1, \text{ if } x \in [t_k, t_{k+1}), \\ 0, \text{ otherwise.} \end{cases}$$

[1] And with the conditions $f^{(2m-2)}(x) = 0$ at the ends of the interval $[x_1, x_K]$; such splines are called natural splines. Alternative conditions at the ends of the interval are also possible.

$$B_k^{l+1}(x) = \frac{x - t_k}{t_{k+l} - t_k} B_k^l(x) + \frac{t_{k+l+1} - x}{t_{k+l+1} - t_{k+1}} B_{k+1}^l(x), \; l = 1, 2, \ldots$$

where $t_k, k = 1, \ldots, r$ is the set of spline knots which may or may not coincide with the data x_k, and the upper index $l+1$ denotes the degree of B-spline. B-splines form a basis in the space of splines, and the interpolating spline can be expressed as [2]

$$f(x) = \sum_{k=1}^{r} a_k B_k(x).$$

An important property of B-splines is the local support, $B_k(x) \geq 0$ only when $x \in [t_k, t_{k+l+1})$. As a consequence, the matrix of the system of equations $\mathbf{B}$ has a banded structure, with just n co-diagonals, and special solution methods are applied.

A.2 Univariate approximation and smoothing

When the data contain inaccuracies, it is pointless to interpolate these data. Methods of approximation and smoothing are applied, which produce functions f that are regularized in some sense, and fit the data in the least squares, least absolute deviation or some other sense.

Let us consider again some basis $\{B_1, \ldots, B_n\}$, so that approximation is sought in the form

$$f(x) = \sum_{i=1}^{n} a_i B_i(x). \tag{A.1}$$

Now n is not the same as the number of data K. Regularization by restricting n to a small number $n \ll K$ is very typical. When B_i are polynomials, the method is called polynomial regression. The basis functions are usually chosen as a system of orthogonal polynomials (e.g., Legendre or Chebyshev polynomials) to ensure a well conditioned system of equations.

The functions B_i can be chosen as B-splines, with a small number of knots t_k fixed in the interval $[x_1, x_K]$. These splines are called regression splines. Regardless what are the basis functions, the coefficients are found by solving the over-determined linear system $\mathbf{Ba} = \mathbf{y}$, with

[2] There are some technical conditions related to the behavior of the spline at the ends of the interval, for instance natural splines require the $(2m-2)$-th derivative at the ends of the interpolation interval be zero. Thus there are more basis functions r than the data, in fact $r = K + 2n$. Also some elements of the B-spline basis require an extended partition, with the first and the last n knots t_k taken outside $[x_1, x_K]$. These technicalities are usually built into the spline algorithms, see [77, 218].

$$\mathbf{B} = \begin{bmatrix} B_1(x_1) & B_2(x_1) & \ldots & B_n(x_1) \\ B_1(x_2) & B_2(x_2) & \ldots & B_n(x_2) \\ \vdots & & & \vdots \\ \vdots & & & \vdots \\ B_1(x_K) & B_2(x_K) & \ldots & B_n(x_K) \end{bmatrix}.$$

Note that the matrix is rectangular, as $n \ll K$, and its rank is usually n. Since not all the equations can be fitted simultaneously, we shall talk about a system of approximate equalities $\mathbf{Ba} \approx \mathbf{y}$.

In the case of least squares approximation, one minimizes the Euclidean norm of the residuals $||\mathbf{Ba} - \mathbf{y}||_2$, or explicitly,

$$\min_{\mathbf{a} \in \Re^n} \left(\sum_{k=1}^{K} (\sum_{i=1}^{n} a_i B_i(x_k) - y_k)^2 \right)^{1/2}. \tag{A.2}$$

In that case there are two equivalent methods of solution. The first method consists in multiplying $\mathbf{B}$ by its transpose and getting the system of *normal equations*

$$\mathbf{Na} = \mathbf{B}^t\mathbf{Ba} = \mathbf{B}^t\mathbf{y}.$$

The entries of an $n \times n$ matrix $\mathbf{N}$ are given as $N_{ij} = \sum_{k=1}^{K} B_i(x_k)B_j(x_k)$. The system of normal equations can also be obtained by writing down the gradient of (A.2) and equalling it to zero.

The alternative is to solve the system $\mathbf{Ba} = \mathbf{y}$ directly using QR-factorization. The pseudo-solution will be precisely the vector $\mathbf{a}$ minimizing the Euclidean norm $||\mathbf{Ba} - \mathbf{y}||$.

An alternative to least squares approximation is the least absolute deviation (LAD) approximation [35]. Here one minimizes

$$\min_{\mathbf{a} \in \Re^n} \sum_{k=1}^{K} |\sum_{i=1}^{n} a_i B_i(x_k) - y_k|, \tag{A.3}$$

possibly subject to some additional constraints discussed later. It is known that the LAD criterion is less sensitive to outliers in the data.

To solve minimization problem (A.3) one uses the following trick to convert it to a linear programming (LP) problem. Let $r_k = f(x_k) - y_k$ be the $k-$ th residual. We represent it as a difference of a positive and negative parts $r_k = r_k^+ - r_k^-$, $r_k^+, r_k^- \geq 0$. The absolute value is $|r_k| = r_k^+ + r_k^-$. Now the problem (A.3) is converted into an LP problem with respect to $\mathbf{a}, \mathbf{r}^+, \mathbf{r}^-$

$$\text{minimize} \qquad \sum_{k=1}^{K} (r_k^+ + r_k^-), \tag{A.4}$$

$$\text{s.t.} \sum_{i=1}^{n} a_i B_i(x_k) - (r_k^+ - r_k^-) = y_k, \; k = 1, \ldots, K$$

$$r_k^+, r_k^- \geq 0.$$

The solution is performed by using the simplex method, or by a specialized version of the simplex method [35]. It is important to note that the system of linear constraints typically has a sparse structure, therefore for large K the use of special programming libraries that employ sparse matrix representation is needed.

Regression splines employ a smaller number of the basis functions than the number of data. An alternative method of spline approximation is using smoothing splines. Natural smoothing splines are solutions to the following problem

$$\text{Minimize } F(f) = p \int_{x_1}^{x_K} |f^{(m)}(t)|^2 dt + \sum_{k=1}^{K} u_k (f(x_k) - y_k)^2,$$

where the smoothing parameter p controls the balance between the requirements of smoothness and fitting the data (with $p = 0$ the solution becomes an interpolating spline, and with $p \to \infty$ it becomes a single polynomial of degree $m - 1$). The values of the weights u_k express the relative accuracy of the data y_k. Smoothing splines are expressed in an appropriate basis (usually B-splines) and the coefficients $\mathbf{a}$ are found by solving a $K \times K$ system of equations with a banded matrix ($2m + 1$ co-diagonals). Details are given in [159].

A.3 Approximation with constraints

Consider a linear least squares or least absolute deviation problem, in which together with the data, additional information is available. For example, there are known bounds on the function f, $L(x) \le f(x) \le U(x)$, or f is known to be monotone increasing, convex, either on the whole of its domain, or on given intervals. This information has to be taken into account when calculating the coefficients $\mathbf{a}$, otherwise the resulting function may fail to satisfy these requirements (even if the data are consistent with them). This is the problem of constrained approximation.

If the constraints are non-linear, then the problem becomes a nonlinear (and sometimes global) optimization problem. This is not desirable, as numerical solutions to such problems could be very expensive. If the problem could be formulated in such a way that the constraints are linear, then it can be solved by standard quadratic and linear programming methods.

Constraints on coefficients

One typical example is linear least squares or least absolute deviation approximation, with the constraints on coefficients $a_i \ge 0$, $\sum_{i=1}^{n} a_i = 1$. There are multiple instances of this problem in our book, when determining the weights of various inputs (in the multivariate setting). We have a system of linear

constraints, and in the case of the least squares approximation, we have the problem

$$\begin{aligned} \text{minimize} \ & \sum_{k=1}^{K} \Big(\sum_{i=1}^{n} a_i B_i(x_k) - y_k\Big)^2 \\ \text{s.t.} \quad & \sum_{i=1}^{n} a_i = 1, a_i \geq 0. \end{aligned} \tag{A.5}$$

It is easy to see that this is a quadratic programming problem, see Section A.5. It is advisable to use standard QP algorithms, as they have proven convergence and are very efficient.

There is an alternative to define the new unrestricted variables using

$$b_i = \frac{\log a_{i+1}}{\log a_i}, i = 1, \ldots, n-1. \tag{A.6}$$

The unconstrained nonlinear optimization problem is solved in the variables b_i, and then the original variables are retrieved using the inverse transformation

$$\begin{aligned} a_1 &= \frac{1}{Z} \\ a_2 &= a_1 e^{b_1} \\ a_3 &= a_1 e^{b_1 + b_2} \\ &\vdots \\ a_n &= a_1 e^{\sum_{i=1}^{n-1} b_i}, \end{aligned} \tag{A.7}$$

with $Z = 1 + \sum_{i=1}^{n-1} e^{\sum_{j=1}^{i} b_j}$.

Unfortunately, the quadratic structure of the least squares problem is lost, the new problem in variables $\mathbf{b}$ is a multiextremal global optimization problem (see Sections A.5.4, A.5.5), which is hard to solve. Nonlinear local optimization methods can be applied, but they do not lead to the globally optimal solution.

In the case of the least absolute deviation, it is quite easy to modify linear programming problem (A.4) to incorporate the new constraints on the variables a_i.

The mentioned constrained linear least squares or least absolute deviation problems are often stated as follows:

$$\begin{aligned} \text{Solve } \mathbf{Ax} &\approx \mathbf{b} \\ \text{s.t. } \mathbf{Cx} &= \mathbf{d} \\ \mathbf{Ex} &\leq \mathbf{f}, \end{aligned} \tag{A.8}$$

where $\mathbf{A}, \mathbf{C}, \mathbf{E}$ are matrices of size $k \times n$, $m \times n$ and $p \times n$, and $\mathbf{b}, \mathbf{d}, \mathbf{f}$ are vectors of length k, m, p respectively. The solution to the system of approximate inequalities is performed in the least squares or least absolute deviation sense

[35, 152]. Here vector $\mathbf{x}$ plays the role of the unknown coefficients $\mathbf{a}$. There are specially adapted versions of quadratic and linear programming algorithms, see Section A.6.

Monotonicity constraints

Another example of constrained approximation problem is monotone (or isotone) approximation. Here the approximated function f is known to be monotone increasing (decreasing), perhaps on some interval, and this has to be incorporated into the approximation process. There are many methods devoted to univariate monotone approximation, most of which are based on spline functions [66, 77, 218].

For regression splines, when using B-spline representation, the problem is exactly of the type (A.5) or (A.8), as monotonicity can be expressed as a set of linear inequalities on spline coefficients. It is possible to choose a different basis in the space of splines (called T-splines [13, 15], they are simply linear combinations of B-splines), in which the monotonicity constraint is expressed even simpler, as non-negativity (or non-positivity) of spline coefficients. It is solved by a simpler version of problem (A.8), called Non-Negative Least Squares (NNLS) [120, 152].

Alternative methods for interpolating and smoothing splines are based on insertion of extra knots (besides the data x_i) and solving a convex nonlinear optimization problem. We mention the algorithms by Schumaker [219] and McAllister and Roulier [177] for quadratic interpolating splines, and by Anderson and Elfving [7] for cubic smoothing splines. Figures A.1, A.2, illustrate various monotone splines.

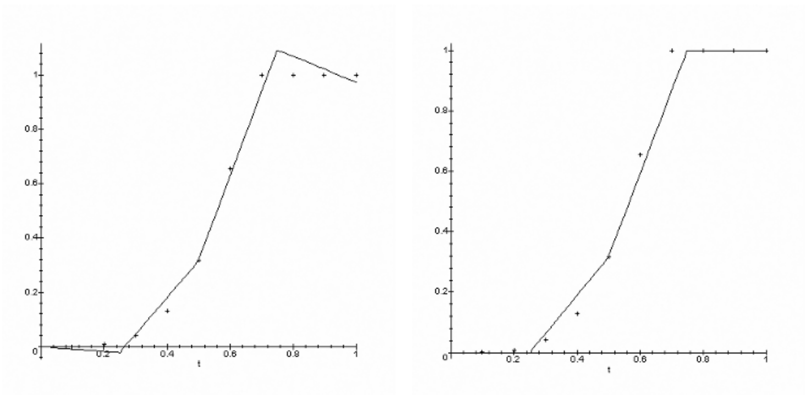


Fig. A.1. Plots of linear regression splines. The spline on the left is not monotone, even though the data are. The spline on the right has monotonicity constraints imposed.

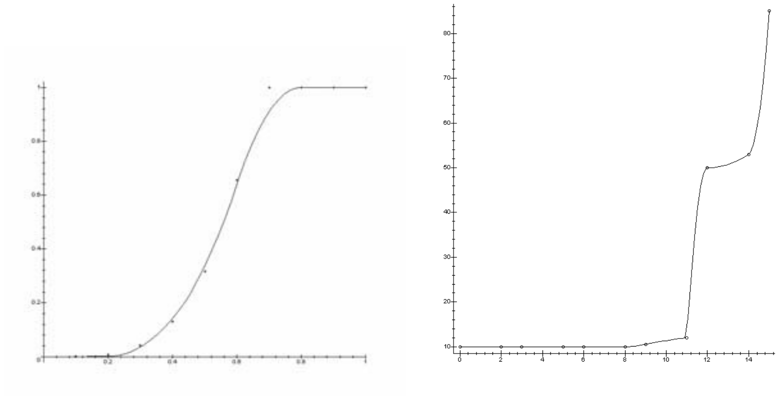


Fig. A.2. Plots of quadratic monotone regression splines.

Convexity constraints

Convexity/concavity of the approximation is another frequently desired property. Convexity can be imposed together with monotonicity. For regression splines, convexity can also be expressed as a set of linear conditions on spline coefficients [13, 15]. This makes convex spline approximation problem formulated as problem (A.8), which is very convenient. Works on constrained interpolating and smoothing splines include [53, 54, 61, 91, 134, 192, 216].

A.4 Multivariate approximation

Linear regression

Linear regression is probably the best known method of multivariate approximation. It consists in building a hyperplane which fits the data best in the least squares sense. Let the equation of the hyperplane be

$$f(\mathbf{x}) = a_0 + a_1 x_1 + a_2 x_2 + \ldots + a_n x_n.$$

Then the vector of coefficients $\mathbf{a}$ can be determined by solving the least squares problem

$$\text{minimize} \sum_{k=1}^{K} (a_0 + \sum_{i=1}^{n} a_i x_{ik} - y_k)^2,$$

where x_{ik} is the i-th component of the vector $\mathbf{x}_k$. Linear regression problem can be immediately generalized if we choose

$$f(\mathbf{x}) = a_0 + \sum_{i=1}^{n} a_i B_i(x_i),$$

where B_i are some given functions of the i-th component of $\mathbf{x}$. In fact, one can define more than one function B_i for the i-th component (we will treat this case below). Then the vector of unknown coefficients can be determined by solving $\mathbf{Ba} \approx \mathbf{y}$ in the least squares sense, in the same way as for the univariate functions described on p. 308. The solution essentially can be obtained by using QR-factorization of $\mathbf{B}$.

It is also possible to add some linear constraints on the coefficients $\mathbf{a}$, for example, non-negativity. Then one obtains a constrained least squares problem, which is an instance of QP. By choosing to minimize the least absolute deviation instead of the least squares criterion, one obtains an LAD problem, which is converted to LP. Both cases can be stated as problem (A.8) on p. 310.

Tensor product schemata

Evidently, linear regression, even with different sets of basis functions B_i, is limited to relatively simple dependence of y on the arguments $\mathbf{x}$. A more general method is to represent a multivariate function f as a tensor product of univariate functions

$$f_i(x_i) = \sum_{j=1}^{J_i} a_{ij} B_{ij}(x_i).$$

Thus each univariate function f_i is written in the form (A.1).

Now, take a product $f(\mathbf{x}) = f_1(x_1)f_2(x_2)\dots f_n(x_n)$. It can be written as

$$f(\mathbf{x}) = \sum_{m=1}^{J} b_m B_m(\mathbf{x}),$$

where $b_m = a_{1j_1}a_{2j_2}\dots a_{nj_n}$, $B_m(\mathbf{x}) = B_{1j_1}(x_1)B_{2j_2}(x_2)\dots B_{nj_n}(x_n)$ and $J = J_1 J_2 \dots J_n$. In this way we clearly see that the vector of unknown coefficients (of length J) can be found by solving a least squares (or LAD) problem (A.2), once we write down the components of the matrix $\mathbf{B}$, namely $\mathbf{B}_{km} = B_m(\mathbf{x}_k)$.

In addition, we can add restrictions on the coefficients, and obtain a constrained LS or LAD problem (A.8). So in principle one can apply the same method of solution as in the univariate case. The problem with the tensor product approach is the sheer number of basis functions and coefficients. For example, if one uses tensor product splines (i.e., B_{ij} are univariate B-splines), say $J_1 = J_2 = \dots J_n = 5$ and works with the data in $\Re^5$, there are $5^5 = 3125$ unknown coefficients. So the size of the matrix $\mathbf{B}$ will be $K \times 3125$. Typically one needs a large number of data $K > J$, otherwise the system is ill-conditioned. Furthermore, depending on the choice of B_{ij}, these data need to be appropriately distributed over the domain (otherwise we may get entire zero columns of $\mathbf{B}$). The problem quickly becomes worse in higher dimensions – a manifestation of so-called curse of dimensionality.

Thus tensor product schemata, like tensor splines, are only applicable in a small dimension when plenty of data is available. For tensor product regression splines, the data may be scattered, but for interpolating and smoothing splines it should be given on a rectangular mesh, which is even worse. Hence they have practical applicability only in two-dimensional case.

Inclusion of monotonicity and convexity constraints for tensor product regression splines is possible. In a suitably chosen basis (like B-splines or T-splines in [13]), monotonicity (with respect to each variable) is written as a set of linear inequalities. Then the LS or LAD approximation becomes the problem (A.8), which is solved by either quadratic or linear programming techniques (see Section A.6). These methods handle well degeneracy of the matrices (e.g., when $K < J$), but one should be aware of their limitations, and general applicability of tensor product schemata to small dimension.

RBF and Neural Networks

These are two popular methods of multivariate nonlinear approximation. In the case of the Radial Basis Functions (RBF) [39], one uses the model

$$f(\mathbf{x}) = \sum_{k=1}^{K} a_k g(||\mathbf{x} - \mathbf{x}_k||),$$

i.e., the same number of basis functions $g_k = g(||\cdot - \mathbf{x}_k||)$ as the data. They are all translations of a single function g, which depends on the radial distance from each datum $\mathbf{x}_k$. Popular choices are thin plate splines, gaussian and multiquadratics.

The function g is chosen so that it decreases with the argument, so that data further away from $\mathbf{x}$ have little influence on the value of $f(\mathbf{x})$. The coefficients are found by solving a least squares problem similar to (A.2). The matrix of this system is not sparse, and the system is large (for large K). Special solution methods based on far-field expansion have been developed to deal with the computational cost associated with solving such systems of equations. Details are given in [39].

We are unaware of any special methods which allow one to preserve monotonicity of f when using RBF approximation.

The Artificial Neural Networks (ANN) is a very popular method of approximation, which has a nice parallel with functioning of neurons, see e.g., [121, 173]. Here a typical approximation model (for a two-layer system) is

$$f(\mathbf{x}) = h\left(\sum_{i=1}^{m} w_i h\left(\sum_{j=1}^{n} u_j x_j + u_0\right) + w_0\right),$$

where h is the transfer function (a sigmoid-type function like $\tan^{-1}$), m is the number of hidden neurons, and w_i, u_j are the weights to be determined from the data. Sometimes x_j is replaced with some value $g(x_j)$.

Training of an ANN consists in identifying the unknown weights w_i, u_j from the data, using essentially the same least squares criterion (although other fitting criteria are often used). Because of nonlinearity in h, this is no longer a quadratic (or even convex) programming problem. While for the output layer weights w_i, this is less problematic [3], for the hidden layer weights u_j it is a non-convex multiextrema optimization problem, which requires global optimization methods. In practice training is done by a crude gradient descent method (called back-propagation), possibly using multistart approach. But even a suboptimal set of weights delivers an adequate precision, and it is argued that more accurate weight determination leads to "overfitting", when the ANN predicts well the training data, but not other values of f.

Among other multivariate approximation scheme we mention the k-nearest neighbors approximation, Sibson's natural neighbor approximation and splines built on triangulations. These methods are described elsewhere [4, 122, 225]. To our knowledge, only a few methods (mostly based on splines) preserve monotonicity of the data.

Lipschitz approximation

This is a new scattered data interpolation and approximation technique based on the Lipschitz properties of the function f [21, 23]. Since Lipschitz condition is expressed as

$$|f(\mathbf{x}) - f(\mathbf{y})| \leq M||\mathbf{x} - \mathbf{y}||,$$

with $||\cdot||$ being some norm, it translates into the tight lower and upper bounds on any Lipschitz function with Lipschitz constant M that can interpolate the data,

$$\begin{aligned} \sigma_u(\mathbf{x}) &= \min_{k=1,\dots,K} \{y_k + M||\mathbf{x} - \mathbf{x}_k||\}, \\ \sigma_l(\mathbf{x}) &= \max_{k=1,\dots,K} \{y_k - M||\mathbf{x}_k - \mathbf{x}||\}. \end{aligned}$$

Then the best possible interpolant in the worst case scenario is given by the arithmetic mean of these bounds. Calculation of the interpolant is straightforward, and no solution to any system of equations (or any training) is required.

The method also works with monotonicity constraints, by using the bounds

$$\begin{aligned} \sigma_u(\mathbf{x}) &= \min_{k} \{y_k + M||(\mathbf{x} - \mathbf{x}_k)_+||\}, \\ \sigma_l(\mathbf{x}) &= \max_{k} \{y_k - M||(\mathbf{x}_k - \mathbf{x})_+||\}, \end{aligned} \tag{A.9}$$

where $\mathbf{z}_+$ denotes the positive part of vector $\mathbf{z}$: $\mathbf{z}_+ = (\bar{z}_1, \dots, \bar{z}_n)$, with

---

[3] One method is to minimize $\sum_{k=1}^{K} \left(\sum_{i=1}^{m} w_i h \left(\sum_{j=1}^{n} u_j x_{kj} + u_0 \right) + w_0 - h^{-1}(y_k) \right)^2$, the problem with respect to w_i is a QP problem.

$$\bar{z}_i = \max\{z_i, 0\}.$$

In fact many other types of constraints can be included as simple bounds on f, see Chapter 6.

What is interesting about Lipschitz interpolant, is that it provides the best possible solution in the worst case scenario, i.e., it delivers a function which minimizes the largest distance from any Lipschitz function that interpolates the data.

If one is interested in smoothing, then the method of Lipschitz smoothing is applied [21]. It consists in determining the smoothened values of y_k that are compatible with a chosen Lipschitz constant. This problem is set as either a QP or LP problem, depending whether we use the LS or LAD criterion.

This method has been generalized for locally Lipschitz functions, where the Lipschitz constant depends on the values of $\mathbf{x}$, and it works for monotone functions.

A.5 Convex and non-convex optimization

When fitting a function to the data, or determining the vector of weights, one has to solve an optimization problem. We have seen that methods of univariate and multivariate approximation require solving such problems, notably the quadratic and linear programming problems. In other cases, like ANN training, the optimization problem is nonlinear. There are several types of optimization problems that frequently arise, and below we outline some of the methods developed for each type. We consider continuous optimization problems, where the domain of the objective function f is $\Re^n$ or a compact subset.

We distinguish unconstrained and constrained optimization. In the first case the domain is $\Re^n$, in the second case the feasible domain is some subset $X \subset \Re^n$, typically determined by a system of equalities and inequalities. We write

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & g_i(\mathbf{x}) = 0, i = 1, \ldots k, \\ & h_i(\mathbf{x}) \leq 0, i = 1, \ldots m. \end{aligned} \tag{A.10}$$

A special case arises when the functions g_i, h_i are linear (or affine). The feasible domain is then a convex polytope, and when in addition to this the objective function is linear or convex quadratic, then special methods of linear and quadratic programming are applied (see Section A.5.2). They work by exploring the boundary of the feasible domain, where the optimum is located.

We also distinguish convex and non-convex optimization. A convex function f satisfies the following condition,

$$f(\alpha\mathbf{x} + (1-\alpha)\mathbf{y}) \leq \alpha f(\mathbf{x}) + (1-\alpha) f(\mathbf{y}),$$

for all $\alpha \in [0, 1]$ and all $\mathbf{x}, \mathbf{y} \in Dom(f)$. If the inequality is strict, it is called strictly convex. What is good about convex functions is that they have a unique minimum (possibly many minimizers), which is the global minimum of the function. Thus if one could check the necessary conditions for a minimum (called KKT (Karush-Kuhn-Tucker) conditions), then one can be certain that the global minimum has been found. Numerical minimization can be performed by any descent scheme, like a quasi-Newton method, steepest descent, coordinate descent, etc. [37, 73, 94, 198, 202].

If the function is not convex, it may still have a unique minimum, although the use of descent methods is more problematic. A special class is that of log-convex (or T-convex) functions, which are functions f, such that $\tilde{f} = \exp(f)$ (or $\tilde{f} = T(f)$) is convex. They are also treated by descent methods (for instance, one can just minimize $\tilde{f}$ instead of f as the minimizers coincide).

General non-convex functions can have multiple local minima, and frequently their number grows exponentially with the dimension n. This number can easily reach $10^{20} - 10^{60}$ for $n < 30$. While locating a local minimum can be done by using descent methods (called local search in this context), there is no guarantee whatsoever that the solution found is anywhere near the global minimum. With such a number of local minima, their enumeration is practically infeasible. This is the problem of global optimization, treated in Sections A.5.4 and A.5.5. The bad news is that in general global optimization problem is unsolvable, even if the minimum is unique [4].

Whenever it is possible to take advantage of convexity or its variants, one should always do this, as more general methods will waste time by chasing non-existent local minima. On the other hand, one should be aware of the implications of non-convexity, especially the multiple local minima problem, and apply proper global optimization algorithms.

We shall also mention the issue of non-differentiable (or non-smooth) optimization. Most local search methods, like quasi-Newton, steepest descent, conjugate gradient, etc., assume the existence of the derivatives (and sometimes all second order derivatives, the Hessian matrix). Not every objective function is differentiable, for example $f(x) = |x|$, or a maximum of differentiable functions. Calculation of descent direction at those points where f does not have a gradient is problematic. Generalizations of the notion of gradient (like Clarke's subdifferential, or quasi-differential [60, 70]) are applied. What exacerbates the problem is that the local/global minimizers are often those points where f is not differentiable. There are a number of derivative-free methods of non-smooth optimization [9, 37, 205, 206], and we particularly mention the Bundle methods [9, 139, 154].

[4] Take as an example function $f(x) = 1$ for all x except $x = a$, where $f(a) = 0$. Unless we know what is a, there is no chance of finding it by exploring the feasible domain. Even relaxations of this problem allowing for continuity of f are still unsolvable.

A.5.1 Univariate optimization

The classical method of univariate optimization is the Newton's method. It works by choosing an initial approximation x_0 and then iterating the following step

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}.$$

Of course, the objective function f needs to be twice differentiable and the derivatives known explicitly. It is possible to approximate the derivatives using finite difference approximation, which leads to the secant method, and more generally to various quasi-Newton schemata.

Newton's method converges to a local minimum of f, and only if certain conditions are satisfied, typically if the initial approximation is close to the minimum (which is unknown in the first place).

For non-differentiable functions the generalizations of the gradient (subgradient, quasi-gradient [60, 70]) are often used. The minimization scheme is similar to the Newton's method, but the derivative is replaced with an approximation of its generalized version.

Golden section method (on an interval) is another classical method, which does not require approximation of the derivatives. It works for unimodal non-differentiable objective functions, not necessarily convex, by iterating the following steps. Let $[x_1, x_2]$ be the interval containing the minimum. At the first step take $x_3 = x_1 + \tau(x_2 - x_1)$, $x_4 = x_2 - \tau(x_2 - x_1)$, with $\tau = \frac{3-\sqrt{5}}{2} \approx 0.382$, a quantity related to the golden ratio ρ as $\tau = 1 - \rho$.[5] At each iteration the algorithm maintains four points x_k, x_l, x_m, x_n, two of which are the ends of the interval containing the minimum. From these four points choose the point with the smallest value of f, say x_k. Then remove the point furthest from x_k, let it be x_n. Arrange the remaining three points in the increasing order, let it be $x_m < x_k < x_l$, and determine the new point using $x = x_l + x_m - x_k$. The minimum is within $[x_m, x_l]$. The iterations stop when the size of the interval is smaller than the required accuracy δ: $|x_m - x_l| < \delta$.

If the objective function is not convex, there could be multiple local minima, as illustrated on Fig. A.3. Locating the global minimum (on a bounded interval) should be done by using a global optimization method. Grid search is the simplest approach, but it is not the most efficient. It is often augmented with some local optimization method, like the Newton's method (i.e., local optimization method is called from the nodes of the grid as starting points).

If the objective function is known to be Lipschitz-continuous, and an estimate of its Lipschitz constant M is available, Pijavski-Shubert method [203, 223] is an efficient way to find and confirm the global optimum. It works

[5] The golden ratio is the positive number satisfying equation $r^2 = r + 1$. Consider the interval $[0, 1]$. Divide it into two parts using the point r. r is the golden section if $\frac{\text{Length of the whole segment}}{\text{Length of the larger segment}} = \frac{\text{Length of the larger segment}}{\text{Length of the smaller segment}}$

as illustrated on Fig. A.3: a saw-tooth underestimate of the objective function f is built, using

$$H^K(x) = \min_{k=1,\ldots,K} f(x_k) - M|x - x_k|,$$

where K denotes the iteration. The global minimizer of H^K is chosen as the next point to evaluate f, and the underestimate is updated after adding the value $f_K = f(x_K)$ and incrementing $K = K + 1$. The sequence of global minima of the underestimates $H^K, K = 1, 2, \ldots$ is known to converge to the global minimum of f.

Calculation of all local minimizers of H^K (the teeth of the saw-tooth underestimate) is done explicitly, and they are arranged into a priority queue, so that the global minimizer is always at the top. Note that there is exactly one local minimizer of H^K between each two neighboring points x_m, x_n. The computational complexity to calculate (and maintain the priority queue) the minimizers of H^K is logarithmic in K. While this method is not as fast as the Newton's method, it guarantees the globally optimal solution in the case of multiextremal objective functions, and does not suffer from lack of convergence.

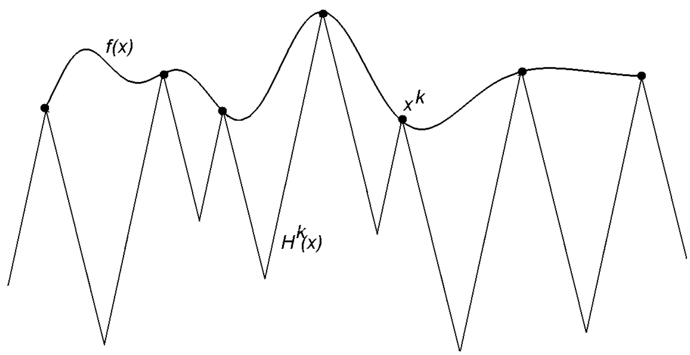


Fig. A.3. Optimization of a non-convex function with many local minima using Pijavski-Shubert method.

A.5.2 Multivariate constrained optimization

Linear programming

As we mentioned, a constrained optimization problem involves constraints on the variables, that may be linear or nonlinear. If the constraints are linear,

and the objective function is also linear, then we have a special case called a linear programming problem (LP). It takes a typical form

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i=1}^{n} c_i x_i = \mathbf{c}^t \mathbf{x} \\
\text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\
& \mathbf{C}\mathbf{x} \leq \mathbf{d} \\
& x_i \geq 0, i = 1, \ldots, n.
\end{aligned}
$$

Here $\mathbf{A}, \mathbf{C}$ are matrices of size $k \times n$, $m \times n$ and $\mathbf{b}, \mathbf{d}$ are vectors of size k and m respectively. Maximization problem is obtained by exchanging the signs of the coefficients c_i, and similarly "greater than" type inequalities are transformed into "smaller than". The condition of non-negativity of x_i can in principle be dropped (such x_i are called unrestricted) with the help of artificial variables, but it is stated as in the standard formulation of LP, and because most solution algorithms assume it by default.

Each LP problem has an associated dual problem (see any textbook on linear programming, e.g., [59, 248]), and the solution to the dual problem allows one to recover that of the primal and vice versa. In some cases solution to the dual problem is computationally less expensive than that to the primal, typically when k and m are large.

The two most used solution methods are the simplex method and the interior point method. In most practical problems, both types of algorithms have an equivalent running time, even though in the worst case scenario the simplex method is exponential and the interior point method is polynomial in complexity.

Quadratic programming

A typical quadratic programming problem is formulated as

$$
\begin{aligned}
\text{minimize} \quad & \tfrac{1}{2}\mathbf{x}^t \mathbf{Q}\mathbf{x} + \mathbf{c}^t \mathbf{x} \\
\text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\
& \mathbf{C}\mathbf{x} \leq \mathbf{d} \\
& x_i \geq 0, i = 1, \ldots, n.
\end{aligned}
$$

Here $\mathbf{Q}$ is a symmetric positive semidefinite matrix (hence the objective function is convex), $\mathbf{A}, \mathbf{C}$ are matrices of constraints, $\mathbf{c}, \mathbf{b}, \mathbf{d}$ are vectors of size n, k, m respectively, and the factor $\frac{1}{2}$ is written for standardization (note that most programming libraries assume it!).

If $\mathbf{Q}$ is indefinite (meaning that the objective function is neither convex nor concave) the optimization problem is extremely complicated because of a very large number of local minima (an instance of an NP-hard problem). If $\mathbf{Q}$ is negative definite, this is the problem of concave programming, which is also

NP-hard. Standard QP algorithms do not treat these cases, but specialized methods are available.

What should be noted with respect to both LP and QP is that the complexity is not in the objective function but in the constraints. Frequently the systems of constraints are very large, but they are also frequently sparse (i.e., contain many 0s). The inner workings of LP and QP algorithms use methods of linear algebra to handle constraints, and special methods are available for sparse matrices. These methods avoid operations with 0s, and perform all operations in sparse matrix format, i.e., when only non-zero elements are stored with their indices. It is advisable to identify sparse matrices and apply suitable methods. If the matrices of constraints are not sparse, then sparse matrix representation is counterproductive.

General constrained nonlinear programming

This is an optimization problem in which the objective function is not linear or quadratic, or constraints $h_i(\mathbf{x}) \leq 0, i = 1, \ldots, m$ are nonlinear. There could be multiple minima, so that this is the problem of global optimization. If the objective function is convex, and the constraints define a convex feasible set, the minimum is unique. It should be noted that even a problem of finding a feasible $\mathbf{x}$ is already complicated.

The two main approaches to constrained optimization are the penalty function and the barrier function approach [198]. In the first case, an auxiliary objective function $\tilde{f}(\mathbf{x}) = f(\mathbf{x}) + \lambda P(\mathbf{x})$ is minimized, where P is the penalty term, a function which is zero in the feasible domain and non-zero elsewhere, increasing with the degree to which the constraints are violated. It can be smooth or non-smooth [202]. λ is a penalty parameter; it is often the case that a sequence of auxiliary objective functions is minimized, with decreasing values of λ. Minimization of $\tilde{f}$ is done by local search methods.

In the case of a barrier function, typical auxiliary functions are

$$\tilde{f}(\mathbf{x}) = f(\mathbf{x}) + \lambda \sum - \ln(-h_i(\mathbf{x})), \quad \tilde{f}(\mathbf{x}) = f(\mathbf{x}) + \lambda \sum (-h_i(\mathbf{x}))^{-r}$$

but now the penalty term is non-zero inside the feasible domain, and grows as $\mathbf{x}$ approaches the boundary.

Recently Sequential Quadratic Programming methods (SQP) have gained popularity for solving constrained nonlinear programming problems, especially those that arise in nonlinear approximation [217]. In essence, this method is based on solving a sequence of QP subproblems at each iteration of the nonlinear optimization problem, by linearizing constraints and approximating the Lagrangian function of the problem (A.10)

$$L(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \sum_{i=1}^{k} \lambda_i g_i(\mathbf{x}) + \sum_{i=m+1}^{k+m} \lambda_i h_{i-k}(\mathbf{x})$$

quadratically (variables λ_i are called the Lagrange multipliers). We refer the reader to [217] for its detailed analysis.

Note that all mentioned methods converge to a locally optimal solution, if f or functions g_i are non-convex. There could be many local optima, and to find the global minimum, global optimization methods are needed.

A.5.3 Multilevel optimization

It is often the case that with respect to some variables the optimization problem is convex, linear or quadratic, and not with respect to the others. Of course one can treat it as a general NLP, but knowing that in most cases we will have a difficult global optimization problem, it makes sense to use the special structure for a subset of variables. This will reduce the complexity of the global optimization problem by reducing the number of variables.

Suppose that we have to minimize $f(\mathbf{x})$ and f is convex with respect to the variables $x_i, i \in \mathcal{I} \subset \{1, 2, \ldots, n\}$, and let $\tilde{\mathcal{I}} = \{1, \ldots, n\} \setminus \mathcal{I}$ denote the complement of this set. Then we have

$$\min_{\mathbf{x}} f(\mathbf{x}) = \min_{x_i : i \in \tilde{\mathcal{I}}} \min_{x_i : i \in \mathcal{I}} f(\mathbf{x}).$$

This is a bi-level optimization problem. At the inner level we treat the variables $x_i, i \in \tilde{\mathcal{I}}$ as constants, and perform minimization with respect to those whose indices are in $\mathcal{I}$. This is done by some efficient local optimization algorithm.

At the outer level we have the global optimization problem

$$\min_{x_i : i \in \tilde{\mathcal{I}}} \tilde{f}(\mathbf{x}),$$

where the function $\tilde{f}$ is the solution to the inner problem. In other words, each time we need a value of $\tilde{f}$, we solve the inner problem with respect to $x_i, i \in \mathcal{I}$.

Of course, the inner problem could be LP (in which case we apply LP methods), or a QP (we apply QP methods). And in principle it is possible to follow this separation strategy and have a multi-level programming problem, where at each level only the variable of a certain kind are treated.

We note that the solution to the inner problem should be the global optimum, not a local minimum. This is the case when the inner problem is LP or QP and the appropriate algorithm is applied.

Sometimes it is possible to have the situation where the function f is convex (say, quadratic positive definite) with respect to both subsets of components $\mathcal{I}$ and $\tilde{\mathcal{I}}$ individually, when the other subset of variables is kept constant. Then one can interchange the inner and outer problems

$$\min_{\mathbf{x}} f(\mathbf{x}) = \min_{x_i : i \in \tilde{\mathcal{I}}} \min_{x_i : i \in \mathcal{I}} f(\mathbf{x}) = \min_{x_i : i \in \mathcal{I}} \min_{x_i : i \in \tilde{\mathcal{I}}} f(\mathbf{x}).$$

However, this does not mean that f is convex with respect to all variables simultaneously. In fact it may be that in each case the outer problem is a multiextrema global optimization problem. It is advisable to use the smallest subset of variables at the outer level.

A.5.4 Global optimization: stochastic methods

Global optimization methods are traditionally divided into two broad categories: stochastic and deterministic [128, 202, 204]. Stochastic methods do not guarantee the globally optimal solution but in probability (i.e., they converge to the global optimum with probability 1, as long as they are allowed to run indefinitively). Of course any algorithm has to be stopped at some time. It is argued that stochastic optimization methods return the global minimum in a finite number of steps with high probability.

Unfortunately there are no rules for how long a method should run to deliver the global optimum with the desired probability, as it depends on the objective function. In some problems this time is not that big, but in others stochastic methods converge extremely slowly, and never find the global solution after any reasonable running time. This is a manifestation of the so–called curse of dimensionality, as the issue is aggravated when the number of variables is increased, in fact the complexity of the optimization problem grows exponentially with the dimension. Even if the class of the objective functions is limited to smooth or Lipschitz functions, the global optimization problem is NP-hard [128, 129, 130, 204].

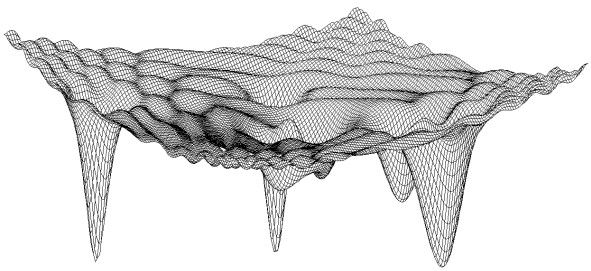


Fig. A.4. An objective function with multiple local minima and stationary points.

The methods in this category include pure random search (i.e., just evaluate and compare the values of f at randomly chosen points), multistart local search, heuristics like simulated annealing, genetic algorithms, tabu search, and many others, see [188, 202]. The choice of the method depends very much on the specific problem, as some methods work faster for certain problem classes. All methods in this category are competitive with one another. There is no general rule for choosing any particular method, it comes down to trial and error.

A.5.5 Global optimization: deterministic methods

Deterministic methods guarantee a globally optimal solution for some classes of objective functions (e.g., Lipschitz functions), however their running time is very large. It also grows exponentially with the dimension, as the optimization problem is NP-hard.

We mention in this category the grid search (i.e., systematic exploration of the domain, possible with the help of local optimization), Branch-and-Bound methods (especially the αBB method [95]), space-filling curves (i.e., representing a multivariate function f through a special univariate function whose values coincide with those of f along an infinite curve which "fills" the domain, see [229]), and multivariate extensions of the Pijavski-Shubert method [16, 18, 119].

One such extension is known as the Cutting Angle methods [212], and the algorithm for the Extended Cutting Angle Method (ECAM) is described in [11, 16, 18, 22]. It mimics the Pijavski-Shubert method in Section A.5.1, although calculation of the local minimizers of the saw-tooth underestimate H^K is significantly more complicated (the number of such minimizers also grows exponentially with the dimension). However in up to 10 variables this method is quite efficient numerically.

We wish to reiterate that there is no magic bullet in global optimization: the general optimization problem is unsolvable, and it is NP-hard in the best case (when restricting the class of the objective functions). It is therefore very important to identify some of the variables, with respect to which the objective function is linear, convex or unimodal, and set up a multilevel optimization problem, as this would reduce the number of global variables, and improve the computational complexity.

A.6 Main tools and libraries

There are a number of commercial and free open source programming libraries that provide efficient and thoroughly tested implementations of the approximation and optimization algorithms discussed in this Appendix. Below is just a sample from an extensive collection of such tools, that in the authors' view are both reliable and sufficiently simple to be used by less experienced users.

Valuable references are `http://plato.asu.edu/sub/nonlsq.html`
`http://www-fp.mcs.anl.gov/otc/Guide/SoftwareGuide`
`http://www2.informs.org/Resources/`

An online textbook on optimization is available from
`http://www.mpri.lsu.edu/textbook/TablCont.htm`

Linear programming

A typical linear programming problem is formulated as

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} c_i x_i = \mathbf{c}^t\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{C}\mathbf{x} \le \mathbf{d} \\ & x_i \ge 0, i = 1, \ldots, n, \end{aligned}$$

where $\mathbf{A}, \mathbf{C}$ are matrices of size $k \times n$, $m \times n$ and $\mathbf{b}, \mathbf{d}$ are vectors of size k and m respectively.

The two most used solution methods are the simplex method and the interior point method [59, 248]. There are a number of standard implementations of both methods. Typically, the user of a programming library is required to specify the entries of the arrays $\mathbf{c}, \mathbf{A}, \mathbf{b}, \mathbf{C}, \mathbf{d}$, point to the unrestricted variables, and sometimes specify the lower and upper bounds on x_i.[6] Most libraries use sparse matrix representation, but they also provide adequate conversion tools.

The packages GLPK (GNU Linear Programming Toolkit) and LPSOLVE `http://www.gnu.org/software/glpk/`
`http://tech.groups.yahoo.com/group/lp_solve/` are both open source and very efficient and reliable. Both implement sparse matrix representation.

Commercial alternatives include CPLEX `http://www.ilog.com`, LINDO `http://www.lindo.com`, MINOS `http://www.stanford.edu/~saunders/brochure/brochure.html` and many others. These packages also include quadratic and general nonlinear programming, as well as mixed integer programming modules.

Quadratic programming

A typical quadratic programming problem is formulated as

$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2}\mathbf{x}^t\mathbf{Q}\mathbf{x} + \mathbf{c}^t\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{C}\mathbf{x} \le \mathbf{d} \\ & x_i \ge 0, i = 1, \ldots, n. \end{aligned}$$

[6] Even though the bounds can be specified through the general constraints in $\mathbf{C}, \mathbf{d}$, in the algorithms the bounds are processed differently (and more efficiently).

Here $\mathbf{Q}$ is a symmetric positive semidefinite matrix (hence the objective function is convex), $\mathbf{A}, \mathbf{C}$ are matrices of constraints, $\mathbf{c}, \mathbf{b}, \mathbf{d}$ are vectors of size n, k, m respectively. Note that most libraries assume the factor $\frac{1}{2}$!

An open source QP solver which supports sparse matrices is OOQP, `http://www.cs.wisc.edu/~swright/ooqp/`. It requires a separate module which should be downloaded from HSL Archive, module MA27, `http://www.cse.clrc.ac.uk/nag/hsl/contents.shtml`

There are many alternative QP solvers, for example Algorithm 559 `http://www.netlib.org/toms/559`, as well as already mentioned commercial CPLEX, LINDO and MINOS packages, see the guides to optimization software `http://plato.asu.edu/sub/nonlsq.html`
`http://www-fp.mcs.anl.gov/otc/Guide/SoftwareGuide`

Least absolute deviation problem

As we mentioned, the LAD problem is converted into an LP problem by using (A.4), hence any LP solver can be used. However there are specially designed versions of the simplex method suitable for LAD problem [35].

The LAD problem is formulated as follows. Solve the system of equations $\mathbf{Ax} \approx \mathbf{b}$ in the least absolute deviation sense, subject to constraints $\mathbf{Cx} = \mathbf{d}$ and $\mathbf{Ex} \leq \mathbf{f}$, where $\mathbf{A}, \mathbf{C}, \mathbf{E}$ and $\mathbf{b}, \mathbf{d}, \mathbf{f}$ are matrices and vectors defined by the user. The computer code can be found in *netlib* `http://www.netlib.org/` as Algorithm 552 `http://www.netlib.org/toms/552`, see also Algorithm 615, Algorithm 478 and Algorithm 551 in the same library.

For Chebyschev approximation code see Algorithm 495 in *netlib* `http://www.netlib.org/toms/495`.

Note that these algorithms are implemented in FORTRAN. A translation into C can be done automatically by `f2c` utility, and is also available from the authors of this book.

Also we should note that all the mentioned algorithms do not use sparse matrix representation, hence they work well with dense matrices of constraints or when the number of constraints is not large. For large sparse LADs, use the generic LP methods.

Constrained least squares

While general QP methods can be applied to this problem, specialized algorithms are available. The ALgorithm 587 from *netlib* solves the following problem called LSEI (Least Squares with Equality and Inequality constraints).

Solve the system of equations $\mathbf{Ax} \approx \mathbf{b}$ in the least squares sense, subject to constraints $\mathbf{Cx} = \mathbf{d}$ and $\mathbf{Ex} \leq \mathbf{f}$, where $\mathbf{A}, \mathbf{C}, \mathbf{E}$ and $\mathbf{b}, \mathbf{d}, \mathbf{f}$ are matrices and vectors defined by the user. The algorithm handles well degeneracy in the systems of equations/constraints.

The computer code (in FORTRAN) can be downloaded from *netlib* `http://www.netlib.org/toms/587`, and its translation into C is available from the authors.

Nonlinear optimization

As a general reference we recommend the following repositories:
`http://www-fp.mcs.anl.gov/otc/Guide/SoftwareGuide`
`http://www2.informs.org/Resources/`
`http://gams.nist.gov/`
`http://www.mat.univie.ac.at/~neum/glopt/software_l.html`

Univariate global optimization

For convex problems the golden section methods is very reliable, it is often combined with the Newton's method. There are multiple implementations, see the references to nonlinear optimization above.

For non-convex multiextremal objective functions, we recommend Pijavski-Shubert method, it is implemented in GANSO library as the special case of ECAM `http://www.ganso.com.au`

Multivariate global optimization

There are a number of repositories and links at
`http://www.mat.univie.ac.at/~neum/glopt/software_g.html`
`http://gams.nist.gov/`

GANSO library `http://www.ganso.com.au` implements a number of global methods, both deterministic (ECAM) and stochastic (multistart random search, heuristics) and also their combinations. It has C/C++, Fortran, Matlab and Maple interfaces.

Spline approximation

Various implementations of interpolating, smoothing and regression splines (univariate and bivariate) are available from *Netlib* and *TOMS* `http://www.netlib.org/tom`. Monotone univariate and tensor product regression splines are implemented in *tspline* package
`http://www.deakin.edu.au/~gleb/tspline.html`

Another implementation is FITPACK `http://www.netlib.org/fitpack/`

Multivariate monotone approximation

tspline package `http://www.deakin.edu.au/~gleb/tspline.html` implements monotone tensor-product regression splines.

The method of monotone Lipschitz approximation is available from LibLip library `http://www.deakin.edu.au/~gleb/lip.html`, and also `http://packages.debian.org/stable/libs/liblip2`

B

Problems

Problems for Chapter 1

Problem B.1. Prove the statement in Note 1.30 on p.13 (Hint: compare to Note 1.25).

Problem B.2. Give other examples of continuous but not Lipschitz continuous aggregation functions, as in Example 1.64 on p.23.

Problem B.3. Write down the dual product in Example 1.77, p. 27, explicitly for two and three arguments.

Problem B.4. Write down the Einstein sum in Example 1.79, p. 27, for three and four arguments (note that this is an associative function).

Problem B.5. Consider Example 1.88 on p. 30. Show that $g(x_1, 1) = \frac{4x_1^2}{(1+x_1)^2} \leq x_1$ for $x_1 \in [0, 1]$.

Problem B.6. Suppose you have the input vector $\mathbf{x} = (0.1, 0.1, 0.5, 0.8)$. Calculate:

1. The arithmetic, geometric and harmonic means;
2. Median and a-Median with $a = 0.5$;
3. OWA with the weighting vector $\mathbf{w} = (0.5, 0.2, 0, 0.3)$;
4. Weighted arithmetic mean $M_{\mathbf{w}}$ with the same weighting vector;
5. Product T_P, dual product S_P, Lukasiewicz t-norm T_L and t-conorm S_L;
6. The $3 - \Pi$ function.

Problem B.7. Show that the function

$$f(x, y) = \frac{x^2 + y^2}{x + y} \text{ if } x + y \neq 0 \text{ and } f(0, 0) = 0,$$

is not an aggregation function. (Hint: you need to check the two main properties of aggregation, and to check monotonicity use the restriction of this function to the edges of the unit square. You may graph the resulting four functions, or compare their values at certain points).

Problem B.8. Express formally the continuity condition and the Lipschitz condition of any aggregation function. Could you give examples for both cases?

Problem B.9. Show that the following function is a strong negation on $[0,1]$ $N(t) = \frac{1-t}{1+\lambda t}$ for $\lambda > -1$.

Problems for Chapter 2

Problem B.10. Let M, G, H and Q be the arithmetic, geometric, harmonic and quadratic means respectively. Also for an aggregation function F let $F_\phi(\mathbf{x}) = \phi^{-1}(F(\phi(x_1), \ldots, \phi(x_n)))$. Prove the following statements:

1. If $\phi : [0,1] \to [0,1]$ is given by $\phi(x) = x^2$ then $M_\phi = Q$ and $G_\phi = G$;
2. If $\phi : [0,1] \to [0,1]$ is given by $\phi(x) = e^{\frac{x-1}{x}}$ then $G_\phi = H$;
3. If $\phi : [0,1] \to [0,1]$ is given by $\phi(x) = 1 - x$ then $M_\phi = M$.

Problem B.11. Show that $A(x_1, \ldots, x_n) = \sum\limits_{i=1}^{n} \frac{2i-1}{n^2} x_i$ is a bisymmetric aggregation function which is neither symmetric nor associative.

Problem B.12. Show that $\mu_k(A) = \left(\frac{|A|}{n}\right)^k$ for some $k > 1$ is a symmetric balanced fuzzy measure, and determine the corresponding OWA function.

Problem B.13. Given the weighting vector $\mathbf{w} = (0.1, 0.2, 0.1, 0, 0.6)$, calculate the orness measure of the OWA function and of the weighted arithmetic mean. Then calculate the OWA function $OWA_\mathbf{w}$ and the weighted mean $M_\mathbf{w}$ of the input vector $\mathbf{x} = (0.4, 0.1, 0.2, 0.6, 0.9)$.

Problem B.14. Given a fuzzy quantifier $Q(t) = t^2$, calculate the weights of the OWA function of dimension $n = 5$. Then calculate its orness measure and the value of this OWA function at $\mathbf{x} = (0.4, 0.1, 0.2, 0.6, 0.9)$.

Problem B.15. Given the fuzzy measure

$$\begin{aligned}
v(\{i\}) &= 0.1,\ i = 1, \ldots, 4;\\
v(\{1,2\}) &= v(\{1,2,3\}) = v(\{1,2,4\}) = 0.5;\\
v(\{1,3\}) &= v(\{1,4\}) = v(\{2,3\}) = v(\{2,4\}) = 0.4;\\
v(\{3,4\}) &= v(\{2,3,4\}) = v(\{1,3,4\}) = v(\{1,2,3,4\}) = 1,
\end{aligned}$$

calculate the Choquet integral of the input vector $\mathbf{x} = (0.4, 0.1, 0.2, 0.5)$.

Problem B.16. Given the λ-fuzzy measure on $2^\mathcal{N}, \mathcal{N} = \{1,2,3\}$, determined by

$$v(\{1\}) = 0.1, v(\{2\}) = 0.1, v(\{3\}) = 0.4,$$

calculate the value of λ (you may need to solve equation (2.71) numerically), and the rest of the coefficients of this fuzzy measure. Then calculate the Choquet integral of the input vector $\mathbf{x} = (0.4, 0.1, 0.2)$.

Problem B.17. Let $\mathcal{N} = \{1,2,3,4\}$. Determine which of the following set functions are fuzzy measures, and explain why

1. $v(\mathcal{A}) = \sum_{i\in\mathcal{A}} i$ for all $\mathcal{A} \subseteq \mathcal{N}$;
2. $v(\mathcal{A}) = \begin{cases} 1, & \text{if } \mathcal{A} \neq \emptyset, \\ 0, & \text{if } \mathcal{A} = \emptyset; \end{cases}$
3. $v(\mathcal{A}) = \sum_{i\in\mathcal{A}} \frac{i}{10}$ for all $\mathcal{A} \subseteq \mathcal{N}$;
4. $v(\mathcal{A}) = \begin{cases} 1, & \text{if } \mathcal{A} = \mathcal{N}, \\ 0, & \text{if } \mathcal{A} = \emptyset, \\ 1/3 & \text{otherwise;} \end{cases}$
5. $v(\mathcal{A}) = \sum_{i\in\mathcal{A}} i - (\frac{|\mathcal{A}|^2}{2})$ for all $\mathcal{A} \subseteq \mathcal{N}$.

Problem B.18. Check for each of the following set functions whether they are λ–fuzzy measures. When they are, determine λ.

1. $\mathcal{N} = \{1,2\}$ and $v(\{1\}) = 1/2, v(\{2\}) = 3/4, v(\emptyset) = 0, v(\mathcal{N}) = 1$;
2. $\mathcal{N} = \{1,2\}$ and $v(\{1\}) = 1/2, v(\{2\}) = 1/3, v(\emptyset) = 0, v(\mathcal{N}) = 1$;
3. Let $\mathcal{N} = \{1,2,3\}$ and $v(\mathcal{A}) = \begin{cases} 1, & \text{if } \mathcal{A} = \mathcal{N}, \\ 0, & \text{if } \mathcal{A} = \emptyset, \\ 1/2 & \text{otherwise} \end{cases}$ for all $\mathcal{A} \subseteq \mathcal{N}$;
4. Let $\mathcal{N} = \{1,2,3\}$ and $v(\mathcal{A}) = \begin{cases} 1 & \text{if } \mathcal{A} = \mathcal{N}, \\ 0 & \text{otherwise} \end{cases}$ for all $\mathcal{A} \subseteq \mathcal{N}$.

Problem B.19.

1. Let v_1 and v_2 be fuzzy measures on $\mathcal{N}$. Show that the set function v defined as $v(\mathcal{A}) = \frac{1}{2}(v_1(\mathcal{A}) + v_2(\mathcal{A}))$ for all $\mathcal{A} \subseteq \mathcal{N}$ is also a fuzzy measure.
2. Prove that the set of all fuzzy measures is convex.[1]

Problem B.20. Show that the set of all self–dual measures is convex. Prove the same statement for the set of all additive measures.

Problem B.21. Prove that the aggregation (by means of any aggregation function) of fuzzy measures is in turn a fuzzy measure, that is, prove the following statement:

Let $f : [0,1]^m \to [0,1]$ be an aggregation function and let $v_1, \ldots, v_m : 2^{\mathcal{N}} \to [0,1]$ be fuzzy measures. Then the set function $v = f(v_1, \ldots, v_m) : 2^{\mathcal{N}} \to [0,1]$, defined for any $\mathcal{A} \subseteq \mathcal{N}$ as

$$v(\mathcal{A}) = f(v_1(\mathcal{A}), \ldots, v_m(\mathcal{A}))$$

is a fuzzy measure.

[1] We remind that a set E is convex if $\alpha x + (1-\alpha)y \in E$ for all $x, y \in E, \alpha \in [0,1]$.

Problem B.22. Let v be $\{0,1\}$–fuzzy measure on $\mathcal{N} = \{1,2,3\}$ given by

$$v(\emptyset) = 0,\ v(\mathcal{N}) = 1,\ v(\{1\}) = v(\{2\}) = v(\{3\}) = 0,$$

$$v(\{1,2\}) = v(\{1,3\}) = 0,\ v(\{2,3\}) = 1.$$

Determine its dual fuzzy measure v^*. After that, show that v is superadditive. Determine whether v^* is subadditive or not.

Problem B.23. Are the following statements true or false:

1. If a measure is balanced then its dual measure is also balanced.
2. If a measure is symmetric then its dual measure is also symmetric.

Problem B.24. Let v be a fuzzy measure and $\mathcal{A}, \mathcal{B}$ be subsets in its domain. Show that:

1. $v(\mathcal{A} \bigcap \mathcal{B}) \leq \min(v(\mathcal{A}), v(\mathcal{B}))$;
2. $v(\mathcal{A} \bigcup \mathcal{B}) \geq \max(v(\mathcal{A}), v(\mathcal{B}))$.

Problem B.25. Show that:

1. $Pos(\mathcal{A}) = \max\limits_{i \in \mathcal{A}}(Pos(\{i\}))$ for all $\mathcal{A} \subseteq \mathcal{N}$;
2. $Nec(\mathcal{A}) = \min\limits_{i \notin \mathcal{A}}(Nec(\mathcal{N} \setminus \{i\}))$ for all $\mathcal{A} \subseteq \mathcal{N}$.

Problem B.26. Let $(Pos\{1\}, Pos\{2\}, Pos\{3\}, Pos\{4\}, Pos\{5\}) = (1, 0.5, 0.3, 0.2, 0.6)$ be a possibility measure on $\mathcal{N} = \{1,2,3,4,5\}$.

1. Determine the value of the possibility measure for each subset of $\mathcal{N}$.
2. Determine the dual of the possibility measure obtained in part 1.

Problem B.27. Prove the two following statements (assuming $|\mathcal{N}| > 2$):

1. Let f be a weighted mean. If $Prob_1, \ldots, Prob_m$ are probability measures, then $v = f(Prob_1, \ldots, Prob_m)$ is a probability measure.
2. Let f be given by $f(x_1, \ldots, x_m) = \max\limits_{i=1,\ldots,m}(f_i(x_i))$, where $f_i : [0,1] \to [0,1]$ are non-decreasing functions satisfying $f_i(0) = 0$ for all $i \in \{1, \ldots, m\}$ and $f_i(1) = 1$ for at least one i. If $Pos_1, \ldots, Pos_m$ are possibility measures, then $v = f(Pos_1, \ldots, Pos_m)$ is a possibility measure.

Problem B.28. Let v be an additive fuzzy measure. Show that the measure of a set $\mathcal{A}$ is determined by the measure of its singletons, i.e., $v(\mathcal{A}) = \sum\limits_{a \in \mathcal{A}} v(\{a\})$.

Problem B.29. Determine discrete Choquet and Sugeno integrals of the vector $\mathbf{x}$ with respect to a fuzzy measure v, where v and $\mathbf{x}$ are given as follows:

1. $\mathcal{N} = \{1, 2\}$ and

$$v(\mathcal{A}) = \begin{cases} 0.5, & \text{if } \mathcal{A} = \{1\} \\ 0, & \text{if } \mathcal{A} = \emptyset \\ 0.7, & \text{if } \mathcal{A} = \{2\} \\ 1, & \text{if } \mathcal{A} = \mathcal{N} \end{cases}$$

   and

$$\text{i) } \mathbf{x} = (0.8, 0.4); \quad \text{ii) } \mathbf{x} = (0.8, 0.9).$$

2. $\mathcal{N} = \{1, 2, 3, 4\}$, v a λ–measure with $v(\{1\}) = 1/15$, $v(\{2\}) = 1/4$, $v(\{3\}) = 1/5, \lambda = 1$ and

$$\text{i) } \mathbf{x} = (2/3, 1/5, 1/2, 1); \quad \text{ii) } \mathbf{x} = (1/2, 1/3, 1/4, 1/5).$$

Problem B.30. Determine the Choquet integral of $\mathbf{x} = (x_1, \ldots, x_4)$ w.r.t the additive fuzzy measure determined by $v = (0.25, 0.35, 0.15, 0.25)$ on $\mathcal{N} = \{1, 2, 3, 4\}$, where each component of v corresponds with the measure on the singleton $\{i\}$ for $i = 1, \ldots, 4$. Which special kind of the Choquet integral-based aggregation function do you obtain?

Problems for Chapter 3

Problem B.31. Prove Proposition 3.20 on p. 132.

Problem B.32. In the discussion of properties of triangular norms on p. 130 we stated that a pointwise minimum or maximum of two t–norms is a) not generally a t–norm, b) it is a conjunctive aggregation function. Prove this statement. You can use a counterexample in part a). Part b) must be a general proof.

Problem B.33. Refer to properties on p. 130. Prove (by example) that not all t–norms are comparable.

Problem B.34. Refer to properties on p. 130. Prove that a linear combination of t–norms $aT_1(\mathbf{x}) + bT_2(\mathbf{x})$, $a, b \in \Re$, is not generally a t–norm (provide a counterexample), although it is a conjunctive extended aggregation function if $a, b \in [0, 1], b = 1 - a$.

Problem B.35. Prove that minimum is a continuous t–norm which is neither strict nor nilpotent (see Section 3.4.3 and Example 3.26).

Problem B.36. Determine multiplicative generators of Schweizer-Sklar, Hamacher, Frank and Yager t–norms and t–conorms, starting from their additive generators.

Problem B.37. Suppose you have the input vector $\mathbf{x} = (0.2, 0.1, 0.5, 0.9)$. Calculate:

1. The Yager, Dombi, Hamacher and Frank t-norms with parameter $\lambda = 2$;
2. The Yager, Dombi, Hamacher and Frank t-conorms with parameter $\lambda = 2$.

Problem B.38. Given the following functions $H_i : [0,1]^2 \to [0,1]$ for $i = 1, \dots, 4$

1. $$H_1(x,y) = \begin{cases} 0, & \text{if } (x,y) \in [0,0.5] \times [0,1], \\ \min(x,y), & \text{otherwise,} \end{cases}$$
2. $$H_2(x,y) = \begin{cases} 0.5, & \text{if } (x,y) \in]0,1[^2, \\ \min(x,y), & \text{otherwise,} \end{cases}$$
3. $$H_3(x,y) = xy \cdot \max(x,y),$$
4. $$H_4(x,y) = 0,$$

determine which ones are a) semicopulas, b) quasi–copulas and c) t–norms.

Problem B.39. Prove the following statements:

1. $T_D(x,y) \le T_L(x,y)$, for all $x, y \in [0,1]$;
2. $T_L(x,y) \le T_P(x,y)$, for all $x, y \in [0,1]$;
3. $T_P(x,y) \le \min(x,y)$, for all $x, y \in [0,1]$;
4. $T(x,y) \le \min(x,y)$, for all $x, y \in [0,1]$ and for any t-norm T;
5. $T_D(x,y) \le T(x,y)$, for all $x, y \in [0,1]$ and for any t-norm T.

Problem B.40. Formulate and prove the statements corresponding to those in Problem B.39 for a t–conorm S.

Problem B.41. Show that:

1. S_L and T_L are dual;
2. S_D and T_D are dual.

Problem B.42. Let T and S be dual t–norms and t–conorms with respect to a strong negation N. Show that the following laws (De Morgan Laws) must hold:

1. $N(T(x,y)) = S(N(x), N(y))$ for all $x, y \in [0,1]$;
2. $N(S(x,y)) = T(N(x), N(y))$ for all $x, y \in [0,1]$.

Problem B.43. Prove that min (max) is the only idempotent t-norm (t-conorm).

Problem B.44. Prove the following statements:

1. A t-conorm S is distributive [2] over a t-norm T if and only if $T = \min$.
2. A t-norm T is distributive over a t-conorm S if and only if $S = \max$.

[2] A bivariate function f is distributive over g if $f(x, g(y,z)) = g(f(x,y), f(x,z))$ and $f(g(x,y),z) = g(f(x,z), f(y,z))$. Note that the second condition is redundant if f is symmetric.

Problem B.45. Let H be a t-norm or a t-conorm. Determine whether $H(x, N(x)) = 0$ and/or $H(x, N(x)) = 1$ for all $x \in [0, 1]$ are valid for the following cases:

1. $H = T_P$ or $H = S_P$;
2. $H = T_L$ or $H = S_L$;
3. $H = T_D$ or $H = S_D$.

Problem B.46. Show that the product t-norm has neither non–trivial idempotent elements, nor zero divisors nor nilpotent elements.

Problem B.47. Show that no element of $]0, 1[$ can be both an idempotent and nilpotent element of a t–norm.

Problem B.48. Show that the set of idempotent elements of the nilpotent minimum t–norm is $\{0\} \bigcup]0.5, 1]$, the set of nilpotent elements is $]0.5, 1]$ and the set of zero divisors is $]0, 1[$.

Problem B.49. Show that the following t–norm is non–continuous

$$T(x, y) = \begin{cases} \frac{xy}{2}, & \text{if } (x, y) \in [0, 1[^2, \\ \min(x, y), & \text{otherwise.} \end{cases}$$

Problem B.50. Justify by means of an example that:

1. An Archimedean t–norm needs not be strictly monotone on $]0, 1]^2$;
2. A strictly monotone (on $]0, 1]^2$) t–norm needs not be a strict t–norm.

Problem B.51. Show that if T is a strict t–norm then it has neither non–trivial idempotent elements nor zero divisors.

Problem B.52. Determine a family of t–norms isomorphic to the product t–norm and another family isomorphic to the Łukasiewicz t–norm.

Problem B.53. Determine the ordinal sum t–norm with summands $(\langle 0, 1/3, T_1\rangle, \langle 2/3, 1, T_2\rangle)$ and T_1, T_2 given by $T_1(x, y) = \max(\frac{x+y+xy-1}{2}, 0), T_2(x, y) = \frac{xy}{x+y-xy}$. Moreover, find the corresponding dual t–conorm.

Problem B.54. Determine the t–norms whose additive generators are:

$$g_1(t) = \begin{cases} 0, & \text{if } t = 1, \\ -\log \frac{t}{2} & \text{otherwise,} \end{cases}$$

and

$$g_2(t) = \begin{cases} 0, & \text{if } t = 1, \\ 3 - t & \text{if } x \in]1/2, 1[, \\ 5 - 2t & \text{otherwise.} \end{cases}$$

Problem B.55. Prove the following statements:

1. A t–norm is a copula if and only if it satisfies the Lipschitz property.

2. Every associative quasi–copula is a copula, and hence it is a t–norm.

Problem B.56. Show that the following functions are copulas if C is a copula:

1. $C_1(x, y) = x - C(x, 1 - y)$;
2. $C_2(x, y) = y - C(1 - x, y)$;
3. $C_3(x, y) = x + y - 1 + C(1 - x, 1 - y)$.

Problem B.57. Show that the following function is a copula
$C_\lambda(x, y) = (\min(x, y))^\lambda \cdot (T_P(x, y))^{1-\lambda}, \quad \lambda \in [0, 1]$.

Problems for Chapter 4

Problem B.58. Prove that uninorms have averaging behavior in the region $[0, e] \times [e, 1] \cup [e, 1] \times [0, e]$ (see p. 201).

Problem B.59. Prove that for any uninorm U it is $U(0, 1) \in \{0, 1\}$.

Problem B.60. Write down the expression of the weakest and the strongest uninorms (Example 4.16 on page 206) for inputs of dimension n.

Problem B.61. Prove that given a representable uninorm U with additive generator u and neutral element $e \in]0, 1[$, the function $N_u(x) = u^{-1}(-u(x))$ is a strong negation with fixed point e and U is self-dual with respect to N_u (excluding the points $(0, 1)$ and $(1, 0)$).

Problem B.62. Check the expression of the MYCIN's uninorm (Example 4.4) rescaled to the domain $[0, 1]$, and prove that T_P and S_P are, respectively, its underlying t-norm and t-conorm.

Problem B.63. Check that the PROSPECTOR's combining function (see Example 4.5), when rescaled to $[0, 1]$, coincides with the 3-Π function given in Example 4.19.

Problem B.64. [99] Given $\lambda > 0$ and the function $u_\lambda : [0, 1] \to [-\infty, +\infty]$ defined by

$$u_\lambda(t) = \log\left(-\frac{1}{\lambda} \cdot \log(1 - t)\right)$$

determine the representable uninorm whose additive generator is u_λ and calculate its neutral element, underlying t-norm and underlying t-conorm.

Problem B.65. Prove that any binary nullnorm $V : [0, 1]^2 \to [0, 1]$ verifies

$$\begin{aligned} \forall t \in [0, a], &\qquad V(t, 1) = a \\ \forall t \in [a, 1], &\qquad V(t, 0) = a \end{aligned}$$

Problem B.66. Using the results of the previous problem, prove the following statement (see page 215), valid for any nullnorm:

$$\forall (x,y) \in [0,a] \times [a,1] \cup [a,1] \times [0,a], \qquad V(x,y) = a$$

Problem B.67. Check that idempotent nullnorms (page 217) are self-dual with respect to any strong negation N with fixed point a.

Problem B.68. Write down the expression of the Łukasiewicz nullnorm (example 4.28 on page 218) for inputs of dimension n.

Problem B.69. Prove that, as stated in Proposition 4.31, generated functions, as given in Definition 4.29, are aggregation functions.

Problem B.70. Check that the generating systems (g_i, h) and $(c \cdot g_i, h \circ \frac{Id}{c})$, $c \in \Re, c > 0$, generate the same function (see Note 4.32).

Problem B.71. Let $\varphi : [0,1] \to [-\infty, +\infty]$ be a continuous strictly increasing function and let $\mathbf{w}$ be a weighting vector. As observed in Section 4.4.1, the generating system given by $g_i(t) = w_i \cdot \varphi(t)$ and $h(t) = \varphi^{-1}(t)$ leads to a quasi-linear mean. Check that the generating system given by $g_i(t) = w_i(a\varphi(t) + b)$ and $h(t) = \varphi^{-1}(\frac{t-b}{a})$, where $a, b \in \Re, a \neq 0$, generates exactly the same quasi-linear mean.

Problem B.72. Prove that a generated function is a weighted mean if and only if it can be generated by a system (g_i, h) such that $g_i(t) = a_i \cdot t + b_i$ and $h(t) = a \cdot t + b$, with $a, a_i, b, b_i \in \Re$, $a, a_i \geq 0$ (See Note 4.33).

Problem B.73. Check (Note 4.35) that quasi-arithmetic means are the only generated functions that are both idempotent and symmetric.

Problem B.74. Use construction on p. 225 in order to build an aggregation function with neutral element $e \in]0,1[$ that behaves as the Łukasiewicz t-norm T_L on $[0,e]^2$ and as the probabilistic sum S_P on $[e,1]^2$. Is the resulting function associative? Is it continuous?

Problem B.75. Check that, as stated in Note 4.52 on page 232, the T-S function generated by $h(t) = a \cdot g(t) + b$, $a, b \in \Re$, $a \neq 0$ coincides with the T-S function generated by g.

Problem B.76. Check the statement on Section 4.5.2 that T-S functions with generating function verifying $g(0) = \pm\infty$ (respectively $g(1) = \pm\infty$) have absorbing element $a = 0$ (respectively $a = 1$).

Problem B.77. Prove (see Section 4.5.2) that the dual of a T-S function $Q_{\gamma,T,S,g}$ with respect to an arbitrary strong negation N is also a T-S function given by $Q_{1-\gamma,S_d,T_d,g_d}$, where S_d is the t-norm dual to S w.r.t. N, T_d is the t-conorm dual to T w.r.t. N and $g_d = g \circ N$.

Problem B.78. Show that, as stated in Note 4.55 on page 234, the function $L_{1/2,T_P,S_P}$ is idempotent for inputs of dimension $n = 2$ but not for inputs of dimension $n = 3$.

Problem B.79. Section 4.5.2 shows the existence of binary idempotent linear convex T-S function built by means of a non-idempotent t-norm and a non-idempotent t-conorm. Prove that this result can be generalized to the class of binary T-S functions by choosing a pair (T, S) verifying the functional equation $(1-\gamma) \cdot g(T(x,y)) + \gamma \cdot g(S(x,y)) = (1-\gamma) \cdot g(x) + \gamma \cdot g(y)$. What kind of averaging functions is obtained?

Problem B.80. Adapt the functional equation given in Problem B.79 to the case of exponential convex T-S functions with parameter $\gamma = 1/2$, and check that the pair (T_0^H, S_P), made of the Hamacher product and the probabilistic sum (see Chapter 3), verifies this equation (i.e., check that the function $E_{1/2,T_0^H,S_P}$ is idempotent when $n = 2$). What happens when $n = 3$?

Problem B.81. Prove that $Q_{\gamma,T,S,N}$ with $N(t) = 1 - t$ (Example 4.57 on page 235) is a linear convex T-S function, and that it is the only T-S function generated by a strong negation which belongs to this class of T-S functions.

Problem B.82. Check that T-S functions generated by $g(t) = \log(N(t))$, where N is a strong negation (see Example 4.58 on page 235) are never exponential convex T-S functions.

Problem B.83. Check that, as stated in the introduction of Section 4.6, the classes of conjunctive and disjunctive functions are dual to each other and that the N-dual of an averaging (respectively mixed) function is in turn an averaging (respectively mixed) function.

Problem B.84. Prove the characterizations of N-symmetric sums given in Propositions 4.63 and 4.67.

Problem B.85. Check that, as stated in Note 4.72, when using Corollary 4.68 to construct symmetric sums, any aggregation function g generates the same symmetric sum as its dual function g_d. Find an example proving that this does not happen in the case of Corollary 4.65.

Problem B.86. Check the statements in Section 4.6.2 regarding N-symmetric sums with absorbing/neutral element, and use them to construct some functions of these types.

Problem B.87. Prove that shift-invariant symmetric sums can be built by means of Corollary 4.68 starting from an arbitrary shift-invariant generating function g.

Problem B.88. Prove the statements in Section 4.7.2 regarding the possession of absorbing and neutral elements of T-OWAs, S-OWAs and ST-OWAs.

Problem B.89. Check (see duality item in Section 4.7.2) that T-OWAs and S-OWAs are dual to each other with respect to the standard negation and that the attitudinal character of a T-OWA and its dual S-OWA are complementary.

Problem B.90. Prove (see duality item in Section 4.7.2) that the class of ST-OWAs is closed under duality.

Problem B.91. Prove the following two statements regarding T-OWAs and S-OWAs:

- A bivariate T-OWA function $O_{T,\mathbf{w}}$ is idempotent if and only if $T = \min$ or $\mathbf{w} = (1, 0)$.
- A bivariate S-OWA function $O_{S,\mathbf{w}}$ is idempotent if and only if $S = \max$ or $\mathbf{w} = (0, 1)$.

References

1. J. Aczél. On mean values. *Bulletin of the American Math. Society*, 54, 1948. 56
2. J. Aczél and C. Alsina. Characterization of some classes of quasilinear functions with applications to triangular norms and to synthesizing judgements. *Methods Oper. Res.*, 48:3–22, 1984. 160
3. B.S. Ahn. On the properties of OWA operator weights functions with constant level of orness. *IEEE Trans. on Fuzzy Systems*, 14:511–515, 2006. 70
4. P. Alfeld. Scattered data interpolation in three or more variables. In L.L. Schumaker and T. Lyche, editors, *Mathematical Methods in Computer Aided Geometric Design*, pages 1–34. Academic Press, New York, 1989. 315
5. C. Alsina. On convex triangle functions. *Aequationes Math.*, 26:91–196, 1983. 183
6. C. Alsina, M.J. Frank, and B. Schweizer. *Associative Functions: Triangular Norms And Copulas*. World Scientific, Singapore, 2006. 126, 127
7. L.-E. Andersson and T. Elfving. Interpolation and approximation by monotone cubic splines. *J. Approx. Theory*, 66:302–333, 1991. 311
8. K.T. Atanassov. *Intuitionistic fuzzy sets*. Physica-Verlag, Heidelberg, New York, 1999. 300
9. A. Bagirov. A method for minimization of quasidifferentiable functions. *Optimization Methods and Software*, 17:31–60, 2002. 267, 317
10. J.F. Banzhaf. Weight voting doesn't work: A mathematical analysis. *Rutgers Law Review*, 19:317–343, 1965. 106
11. L.M. Batten and G. Beliakov. Fast algorithm for the cutting angle method of global optimization. *Journal of Global Optimization*, 24:149–161, 2002. 324
12. I. Batyrshin and O. Kaynak. Parametric classes of generalized conjunction and disjunction operators for fuzzy modeling. *IEEE Trans. on Fuzzy Systems*, 7:586–596, 1999. 186, 187, 189
13. G. Beliakov. Shape preserving approximation using least squares splines. *Approximation Theory and Applications*, 16:80–98, 2000. 61, 84, 89, 173, 175, 212, 272, 311, 312
14. G. Beliakov. Shape preserving splines in constructing WOWA operators: Comment on paper by V. Torra in Fuzzy Sets and Systems 113 (2000) 389-396. *Fuzzy Sets and Systems*, 121:549–550, 2001. 72
15. G. Beliakov. Monotone approximation of aggregation operators using least squares splines. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10:659–676, 2002. 173, 212, 272, 273, 275, 311, 312

16. G. Beliakov. Geometry and combinatorics of the cutting angle method. *Optimization*, 52:379–394, 2003. 324
17. G Beliakov. How to build aggregation operators from data? *Int. J. Intelligent Systems*, 18:903–923, 2003. 59, 61, 77, 89, 175, 272, 273, 275
18. G. Beliakov. The cutting angle method - a tool for constrained global optimization. *Optimization Methods and Software*, 19:137–151, 2004. 324
19. G. Beliakov. Identification of general aggregation operators by Lipschitz approximation. In M.H. Hamza, editor, *The IASTED Intl. Conf. on Artificial Intelligence and Applications*, pages 230–233, Innsbruck, Austria, 2005. ACTA Press. 272, 276
20. G. Beliakov. Learning weights in the generalized OWA operators. *Fuzzy Optimization and Decision Making*, 4:119–130, 2005. 59, 77, 83, 87, 89, 90
21. G. Beliakov. Monotonicity preserving approximation of multivariate scattered data. *BIT*, 45:653–677, 2005. 276, 315, 316
22. G. Beliakov. A review of applications of the cutting angle methods. In A. Rubinov and V. Jeyakumar, editors, *Continuous Optimization*, pages 209–248. Springer, New York, 2005. 324
23. G Beliakov. Interpolation of Lipschitz functions. *J. of Comp. and Applied Mathematics*, 196:20–44, 2006. 315
24. G. Beliakov. Construction of aggregation operators for automated decision making via optimal interpolation and global optimization. *J. of Industrial and Management Optimization*, 3:193–208, 2007. 280, 292, 296
25. G. Beliakov and T. Calvo. Identification of general and double aggregation operators using monotone smoothing. In E. Montseny and P. Sobrevilla, editors, *EUSFLAT 2005*, pages 937–942, Barcelona, Spain, 2005. European Society for Fuzzy Logic and Technology. 272, 276
26. G. Beliakov and T. Calvo. Construction of aggregation operators with noble reinforcement. *IEEE Trans. on Fuzzy Syst.*, in press, doi 10.1109/TFUZZ.2006.890675, 2007. 192, 272, 292, 293, 296
27. G. Beliakov, T. Calvo, and J. Lázaro. Pointwise construction of Lipschitz aggregation operators with specific properties. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 15:193–223, 2007. 272, 276, 280
28. G. Beliakov, T. Calvo, and A. Pradera. Absorbent tuples of aggregation operators. *Fuzzy Sets and Systems*, in press, doi:10.1016/j.fss.2007.03.007, 2007. 13, 279
29. G. Beliakov, T. Calvo, and A. Pradera. Handling of neutral information by aggregation operators. *Fuzzy Sets and Systems*, 158:861–880, 2007. 13, 281
30. G. Beliakov, R. Mesiar, and L. Valaskova. Fitting generated aggregation operators to empirical data. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 12:219–236, 2004. 59, 83, 86
31. G. Beliakov and L. Troiano. Fitting ST-OWA operators to empirical data. In *EUSFLAT 2007*. European Society for Fuzzy Logic and Technology, 2007. 249
32. G. Beliakov and J. Warren. Appropriate choice of aggregation operators in fuzzy decision support systems. *IEEE Trans. On Fuzzy Systems*, 9:773–784, 2001. 272
33. R.E. Bellman and M. Giertz. On the analytic formalism of the theory of fuzzy sets. *Inform. Sci.*, 5:149–156, 1973. 23
34. R.E. Bellman and L. Zadeh. Decisionmaking in a fuzzy environment. *Management Science*, 17:141–164, 1970. 8
35. P. Bloomfield and W.L. Steiger. *Least Absolute Deviations. Theory, Applications and Algorithms*. Birkhauser, Boston, Basel, Stuttgart, 1983. 259, 308, 309, 311, 326

36. C. Bonferroni. Sulle medie multiple di potenze. *Bollettino Matematica Italiana*, 5:267–270, 1950. 63
37. J.F. Bonnans, J.C. Gilbert, C. Lemarechal, and C.A. Sagastizabal. *Numerical Optimization. Theoretical and Practical Aspects*. Springer, Berlin, Heidelberg, 2000. 317
38. B. Buchanan and E. Shortliffe. *Rule-based Expert Systems. The MYCIN Experiments of the Stanford Heuristic Programming Project*. Addison-Wesley, Reading, MA, 1984. 28, 31, 197, 198
39. M.D. Buhmann. *Radial Basis Functions. Theory and Implementations*. Cambridge University Press, Cambridge, 2003. 314
40. P.S. Bullen. *Handbook of Means and Their Inequalities*. Kluwer, Dordrecht, 2003. 24, 39, 40, 50, 62, 299
41. R.L. Burden and J.D. Faires. *Numerical Analysis*. Brooks/Cole, Pacific Grove, CA, 2001. 305
42. T. Calvo, B. De Baets, and J. Fodor. The functional equations of Frank and Alsina for uninorms and nullnorms. *Fuzzy Sets and Systems*, 120:385–394, 2001. 214
43. T. Calvo, A. Kolesárová, M. Komorníková, and R. Mesiar. Aggregation operators: properties, classes and construction methods. In T. Calvo, G. Mayor, and R. Mesiar, editors, *Aggregation Operators. New Trends and Applications*, pages 3–104. Physica-Verlag, Heidelberg, New York, 2002. 8, 30, 39, 73, 227, 239, 241, 297, 2
44. T. Calvo and G. Mayor. Remarks on two types aggregation functions. *Tatra Mount. Math. Publ.*, 16:235–254, 1999. 54, 55, 56
45. T. Calvo and R. Mesiar. Stability of aggregation operators. In *2nd Conf. of the Europ. Soc. for Fuzzy Logic and Technology*, pages 475–478, Leicester, 2001. 272
46. T. Calvo and R. Mesiar. Continuous generated associative aggregation operators. *Fuzzy Sets and Systems*, 126:191–197, 2002. 224, 225
47. T. Calvo and R. Mesiar. Weighted triangular norms-based aggregation operators. *Fuzzy Sets and Systems*, 137:3–10, 2003. 176, 180, 226
48. T. Calvo and R. Mesiar. How to incorporate weights into aggregation process? In K.T. Atanassov, O. Hryniewicz, and J. Kacprzyk, editors, *Soft Computing. Foundations and Theoretical Aspects*, pages 97–122. EXIT Publishing House, Warzsawa, 2004. 176, 226
49. T. Calvo, R. Mesiar, and R. Yager. Quantitative weights and aggregation. *IEEE Trans. on Fuzzy Systems*, 12:62–69, 2004. 176, 226, 299
50. T. Calvo and A. Pradera. Double aggregation operators. *Fuzzy Sets and Systems*, 142:15–33, 2004. 300, 301
51. M. Carbonell, M. Mas, and G. Mayor. On a class of monotonic extended OWA operators. In *6th IEEE International Conference on Fuzzy Systems*, volume III, pages 1695–1700, Barcelona, Spain, 1997. 79
52. R.E. Carlson and F.N. Fritsch. Monotone piecewise bicubic interpolation. *SIAM J. Numer. Anal.*, 22:386–400, 1985. 273
53. J. Carnicer. Multivariate convexity preserving interpolation by smooth functions. *Adv. Comput. Mathem.*, 3:395–404, 1995. 312
54. J. Carnicer and M.S. Floater. Piecewise linear interpolants to Lagrange and Hermite convex scattered data. *Numerical Algorithms*, 13:345–364, 1996. 312
55. S.-J. Chen and C.-L. Hwang. *Fuzzy Multiple Attribute Decision Making: Methods and Applications*. Springer, Berlin, Heidelberg, 1992. 6, 7
56. T.-Y. Chen and J.-C. Wang. Identification of λ-fuzzy measures using sampling design and genetic algorithms. *Fuzzy Sets and Systems*, 123:321–341, 2001. 113

57. E. W. Cheney and D. Kincaid. *Numerical Mathematics and Computing*. Brooks/Cole, Belmont, CA., 2004. 305
58. G. Choquet. Theory of capacities. *Ann. Inst. Fourier*, 5:1953–1954, 1953. 91
59. V. Chvatal. *Linear Programming*. W.H. Freeman, New York, 1983. 320, 325
60. F.H. Clarke. *Optimization and Nonsmooth Analysis*. John Wiley, New York, 1983. 317, 318
61. J.C. Clements. Convexity-preserving piecewise rational cubic interpolation. *SIAM J. Numer. Anal.*, 27:1016–1023, 1990. 312
62. P. Costantini and C. Manni. Monotonicity-preserving interpolation of nongridded data. *Computer Aided Geometric Design*, 13:467–497, 1996. 273
63. P. Costantini and C. Manni. A local shape-preserving interpolation scheme for scattered data. *Computer Aided Geometric Design*, 16:385–405, 1999. 273
64. B. De Baets. Idempotent uninorms. *Europ. J. Oper. Research*, 118:631–642, 1999. 202, 303
65. B. De Baets and J. Fodor. Van Melle's combining function in MYCIN is a representable uninorm: An alternative proof. *Fuzzy Sets and Systems*, 104:133–136, 1999. 200, 209
66. C. De Boor. *A Practical Guide to Splines*. Springer, New York, revised edition, 2001. 306, 311
67. H. De Meyer and B. De Baets. Fitting piecewise linear copulas to data. In *4th Conf. of the Europ. Soc. for Fuzzy Logic and Technology*, Barcelona, Spain, 2005. 272
68. J. Del Amo, J. Montero, and E. Molina. Representation of recursive rules. *Europ. J. Oper. Research*, 130:29–53, 2001. 15
69. M. Delgado, J. L. Verdegay, and M.A. Vila. On aggregation operations of linguistic labels. *Int. J. Intelligent Systems*, 8:351–370, 1993. 300
70. V. F. Demyanov and A. Rubinov. *Quasi-differential Calculus*. Optimization Software, Inc., New York, 1986. 317, 318
71. D. Denneberg. *Non-additive Measure and Integral*. Kluwer, Dordrecht, 1994. 90, 91, 92
72. D. Denneberg. Non–additive measure and integral, basic concepts and their role for applications. In M. Grabisch, T. Murofushi, and M. Sugeno, editors, *Fuzzy Measures and Integrals. Theory and Applications*, pages 42–69. Physica-Verlag, Heidelberg, 2000. 101
73. J.E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, NJ, 1983. 317
74. G. Deschrijver. The Archimedean property for t-norms in interval-valued fuzzy set theory. *Fuzzy Sets and Systems*, 157:2311–2327, 2006. 300
75. G. Deschrijver. Additive and multiplicative generators in interval-valued fuzzy set theory. *IEEE Trans. on Fuzzy Systems*, in press, 2007. 300
76. G. Deschrijver, C. Cornelis, and E.E. Kerre. On the representation of intuitionistic fuzzy t-norms and t-conorms. *IEEE Trans. on Fuzzy Systems*, 12:45–61, 2004. 300
77. P. Dierckx. *Curve and surface fitting with splines*. Clarendon press, Oxford, 1995. 173, 306, 307, 311
78. J. Dombi. Basic concepts for a theory of evaluation: the aggregative operator. *Europ. J. Oper. Research*, 10:282–293, 1982. 200, 208, 209, 240
79. J. Dombi. A general class of fuzzy operators, the De Morgan class of fuzzy operators and fuzziness measures induced by fuzzy operators. *Fuzzy Sets and Systems*, 8:149–163, 1982. 158

80. D. Dubois, J.-L. Marichal, H. Prade, M. Roubens, and R. Sabbadin. The use of the discrete Sugeno integral in decision making: a survey. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9:539–561, 2001. 116
81. D. Dubois and H. Prade. *Fuzzy Sets and Systems. Theory and Applications.* Academic Press, New York, 1980. 8
82. D. Dubois and H. Prade. Criteria aggregation and ranking of alternatives in the framework of fuzzy set theory. In H.J. Zimmerman, L.A. Zadeh, and B.R. Gaines, editors, *TIMS/Studies in the Management Science*, pages 209–240. Elsevier Science Publishers, 1985. 230
83. D. Dubois and H. Prade. A review of fuzzy set aggregation connectives. *Inform. Sci.*, 36:85–121, 1985. 8, 39, 177, 179, 240, 241, 245
84. D. Dubois and H. Prade. Weighted minimum and maximum operations in fuzzy sets theory. *Inform. Sci.*, 39:205–210, 1986. 176
85. D. Dubois and H. Prade. *Possibility Theory.* Plenum Press, New York, 1988. 101
86. D. Dubois and H. Prade. *Fundamentals of Fuzzy Sets.* Kluwer, Boston, 2000. 7, 8, 92
87. D. Dubois and H. Prade. On the use of aggregation operations in information fusion processes. *Fuzzy Sets and Systems*, 142:143–161, 2004. 7, 8, 39
88. R. Duda, P. Hart, and N. Nilsson. Subjective Bayesian methods for rule-based inference systems. In *Proc. Nat. Comput. Conf. (AFIPS)*, volume 45, pages 1075–1082, 1976. 28, 31, 197, 198
89. J.J. Dujmovic. Two integrals related to means. *Univ. Beograd. Publ. Elektrotechn. Fak.*, pages 231–232, 1973. 39, 40
90. J.J. Dujmovic. Weighted conjunctive and disjunctive means and their application in system evaluation. *Univ. Beograd. Publ. Elektrotechn. Fak.*, pages 147–158, 1974. 39, 40
91. T. Elfving and L.-E. Andersson. An algorithm for computing constrained smoothing spline functions. *Numer. Math.*, 52:583–595, 1989. 312
92. J.M. Fernández Salido and S. Murakami. Extending Yager's orness concept for the OWA aggregators to other mean operators. *Fuzzy Sets and Systems*, 139:515–542, 2003. 40
93. D. Filev and R. Yager. On the issue of obtaining OWA operator weights. *Fuzzy Sets and Systems*, 94:157–169, 1998. 77
94. R. Fletcher. *Practical Methods of Optimization.* Wiley, New York, 2nd edition edition, 2000. 317
95. C. A. Floudas. *Deterministic Global Optimization: Theory, Methods, and Applications.* Kluwer, Dordrecht; London, 2000. 324
96. J. Fodor and J.-L. Marichal. On nonstrict means. *Aequationes Mathematicae*, 54:308–327, 1997. 56
97. J. Fodor and M. Roubens. *Fuzzy Preference Modelling and Multicriteria Decision Support.* Kluwer, Dordrecht, 1994. 39, 40, 179
98. J. Fodor and R. Yager. Fuzzy set-theoretic operators and quantifiers. In D. Dubois and H. Prade, editors, *The Handbook of Fuzzy Sets*, volume 7, pages 125–193. Kluwer, Dordrecht, 2000. 179
99. J. Fodor, R. Yager, and A. Rybalov. Structure of uninorms. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 5:411–427, 1997. 208, 225, 336
100. R. Fuller and P Majlender. An analytic approach for obtaining maximal entropy OWA operator weights. *Fuzzy Sets and Systems*, 124:53–57, 2001. 77, 78
101. R. Fuller and P. Majlender. On obtaining minimal variability OWA operator weights. *Fuzzy Sets and Systems*, 136:203–215, 2003. 71, 79, 80

102. J. García-Lapresta and B. Llamazares. Aggregation of fuzzy preferences: some rules of the mean. *Soc. Choice Welfare*, 17:673–690, 2000. 240
103. J. García-Lapresta and L.C. Meneses. Individual-valued preferences and their aggregation: consistency analysis in a real case. *Fuzzy Sets and Systems*, 151:269–284, 2005. 240, 244
104. J. García-Lapresta and R. Marques Pereira. Constructing reciprocal and stable aggregation operators. In *Proc. Internat. Summer School on Aggregation Operators and their Applications*, pages 73–78, Oviedo, Spain, 2003. 240, 242, 244
105. E. M. Gertz and S. J. Wright. Object-oriented software for quadratic programming. *ACM Trans. Math. Software*, 29:58–81, 2003. 273
106. M. Grabisch. The applications of fuzzy integrals in multicriteria decision making. *Europ. J. Operations Research*, 89:445–456, 1996. 91, 97
107. M. Grabisch. k-order additive discrete fuzzy measures and their representation. *Fuzzy Sets and Systems*, 92:167–189, 1997. 93, 105, 106, 107
108. M. Grabisch. The interaction and Möbius representation of fuzzy measures on finite spaces, k-additive measures: a survey. In M. Grabisch, T. Murofushi, and M. Sugeno, editors, *Fuzzy Measures and Integrals. Theory and Applications*, pages 70–93. Physica-Verlag, Heidelberg, 2000. 93, 101, 105, 107
109. M. Grabisch. The Choquet integral as a linear interpolator. In *10th Int. Conf. on Information Processing and Management of Uncertainty*, pages 373–378, Perugia, Italy, 2004. 272
110. M. Grabisch, I. Kojadinovic, and P. Meyer. A review of methods for capacity identification in Choquet integral based multi-attribute utility theory. *Europ. J. Operations Research*, in press, 2007. 91, 111, 112
111. M. Grabisch and C. Labreuche. To be symmetric or asymmetric? A dilemma in decision making. In J. Fodor, B. De Baets, and P. Perny, editors, *Preferences and Decisions under Incomplete Knowledge*. Physica–Verlag, Heidelberg, 2000. 91
112. M. Grabisch and C. Labreuche. Bi-capacities — Part I: definition, Möbius transform and interaction. *Fuzzy Sets and Systems*, 151:211–236, 2005. 299
113. M. Grabisch and C. Labreuche. Bi-capacities — Part II: the Choquet integral. *Fuzzy Sets and Systems*, 151:237–259, 2005. 299
114. M. Grabisch, J.-L. Marichal, R. Mesiar, and E. Pap. *Aggregation Functions*. Elsevier, to appear, 2007. 297
115. M. Grabisch, J.-L. Marichal, and M. Roubens. Equivalent representations of set functions. *Mathematics of Oper. Res.*, 25:157–178, 2000. 106, 107
116. M. Grabisch, T. Murofushi, and M. Sugeno, editors. *Fuzzy Measures and Integrals. Theory and Applications*. Physica-Verlag, Heidelberg, 2000. 91, 92, 102, 111
117. M. Grabisch, H.T. Nguyen, and E.A. Walker. *Fundamentals of Uncertainty Calculi, with Applications to Fuzzy Inference*. Kluwer, Dordrecht, 1995. 103, 108
118. L. Han and L.L. Schumaker. Fitting monotone surfaces to scattered data using C1 piecewise cubics. *SIAM J. Numer. Anal.*, 34:569–585, 1997. 273
119. P. Hansen and B. Jaumard. Lipschitz optimization. In Reiner Horst and P. Pardalos, editors, *Handbook of Global Optimization*, pages 407–493. Kluwer, Dordrecht, 1995. 324
120. K.H. Haskell and R. Hanson. An algorithm for linear least squares problems with equality and nonnegativity constraints. *Mathematical Programming*, 21:98–118, 1981. 311

121. M.H. Hassoun. *Fundamentals of Artificial Neural Networks*. MIT Press, Massachusetts, 1995. 314
122. T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, New York, Berlin, Heidelberg, 2001. 315
123. F. Herrera and E. Herrera-Viedma. Aggregation operators for linguistic weighted information. *IEEE Trans. on Syst., Man, and Cybernetics*, 27:646–656, 1997. 300
124. F. Herrera and E. Herrera-Viedma. Linguistic decision analysis: Steps for solving decision problems under linguistic information. *Fuzzy Sets and Systems*, 115:67–82, 2000. 300
125. F. Herrera and E. Herrera-Viedma. A study of the origin and uses of the Ordered Weighted Geometric operator in multicriteria decision making. *Int. J. Intelligent Systems*, 18:689–707, 2003. 73, 300
126. F. Herrera, E. Herrera-Viedma, and J. L. Verdegay. Direct approach processes in group decision making using linguistic OWA operators. *Fuzzy Sets and Systems*, 79:175–190, 1996. 300
127. F. Herrera, E. Herrera-Viedma, and J. L. Verdegay. Choice processes for non-homogeneous group decision making in linguistic setting. *Fuzzy Sets and Systems*, 94:287–308, 1998. 300
128. R. Horst, P. Pardalos, and N.V. Thoai, editors. *Introduction to Global Optimization*. Kluwer, Dordrecht, 2nd edition, 2000. 323
129. R. Horst and P. M. Pardalos. *Handbook of Global Optimization*. Kluwer, Dordrecht; Boston, 1995. 323
130. R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches*. Springer-Verlag, Berlin; New York, 2nd rev. edition, 1993. 323
131. C.-L. Hwang and M.J. Lin. *Group Decision Making under Multiple Criteria*. Springer, Berlin, Heidelberg, 1987. 7
132. C.-L. Hwang and A.S.M. Masud. *Multiple Objective Decitions Making. Methods and Applications*. Springer, Berlin, Heidelberg, 1979. 6
133. C.-L. Hwang and K. Yoon. *Multiple Attribute Decision Making. Methods and Applications*. Springer, Berlin, Heidelberg, 1981. 6
134. R. Iqbal. A one-pass algorithm for shape-preserving quadratic spline interpolation. *J. Sci. Comput.*, 7:359–376, 1992. 312
135. K. Ishii and M. Sugeno. A model of human evaluation process using fuzzy measure. *Int. Journal of Man-Machine Studies*, 22:19–38, 1985. 113
136. S. Jenei. On archimedian triangular norms. *Fuzzy Sets and Systems*, 99: 179–186, 1998. 148, 174, 212
137. S. Jenei and J. Fodor. On continuous triangular norms. *Fuzzy Sets and Systems*, 100:273–282, 1998. 148, 174, 212
138. U. Kaymak and H.R. van Nauta Lemke. Selecting an aggregation operator for fuzzy decision making. In *3rd IEEE Intl. Conf. on Fuzzy Systems*, volume 2, pages 1418–1422, 1994. 34, 262
139. K.C. Kiwiel. *Methods of Descent of Nondifferentiable Optimization*, volume 1133 of *Lecture Notes in Mathematics*. Springer, Berlin, 1985. 267, 317
140. E.P. Klement and A. Kolesárová. Extension to copulas and quasi-copulas as special 1-Lipschitz aggregation operators. *Kybernetika*, 41:329–348, 2005. 272, 285
141. E.P. Klement, R. Mesiar, and E. Pap. On the relationship of associative compensatory operators to triangular norms and conorms. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 4:129–144, 1996. 204, 206, 208

142. E.P. Klement, R. Mesiar, and E. Pap. *Triangular Norms.* Kluwer, Dordrecht, 2000. 126, 134, 141, 143, 145, 148, 149, 167, 182, 192
143. G. Klir, Z. Wang, and D. Harmanec. Constructing fuzzy measures in expert systems. *Fuzzy Sets and Systems*, 92:251–264, 1997. 113
144. I. Kojadinovic. A weight-based approach to the measurement of the interaction among criteria in the framework of aggregation by the bipolar Choquet integral. *Europ. J. Oper. Res.*, 179:498–517, 2007. 299
145. I. Kojadinovic and M. Grabisch. Non additive measure and integral manipulation functions, R package version 0.2, 2005. 115
146. I. Kojadinovic and J.-L. Marichal. Entropy of bi-capacities. *Europ. J. Oper. Res.*, 178:168–184, 2007. 299
147. I. Kojadinovic, J.-L. Marichal, and M. Roubens. An axiomatic approach to the definition of the entropy of a discrete Choquet capacity. *Inform. Sci.*, 172:131–153, 2005. 108
148. A. Kolesárová, R. Mesiar, and G. Mayor. Weighted ordinal means. *Inform. Sci.*, in press, doi:10.1016/j.ins.2007.03.003, 2007. 121, 300
149. A. Kolesárová, R. Mesiar, J. Mordelová, and C. Sempi. Discrete copulas. *IEEE Trans. on Fuzzy Systems*, 14:608–705, 2006. 300
150. A. Kolesárová, E. Muel, and J. Mordelová. Construction of kernel aggregation operators from marginal values. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10 Suppl.:37–49, 2002. 272, 286
151. M. Komorníková. Aggregation operators and additive generators. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9:205–215, 2001. 48, 222, 223, 224, 2
152. C. Lawson and R. Hanson. *Solving Least Squares Problems.* SIAM, Philadelphia, 1995. 173, 311
153. K.M. Lee and H. Leekwang. Identification of λ-fuzzy measure by genetic algorithms. *Fuzzy Sets and Systems*, 75:301–309, 1995. 113
154. C. Lemarechal. Bundle methods in non-smooth optimization. In C. Lemarechal and R. Mifflin, editors, *Non-smooth Optimization*, pages 78–102. Pergamon Press, Oxford, 1978. 267, 317
155. C.M. Ling. Representation of associative functions. *Publ. Math. Debrecen*, 12:189–212, 1965. 144, 167
156. X. Liu. On the properties of equidifferent RIM quantifier with generating function. *Int. J. General Syst.*, 34:579–594, 2005. 82
157. X. Liu. The solution equivalence of minimax disparity and minimum variance problems for OWA operators. *Int. J. Approx. Reasoning*, 45:68–81, 2007. 80
158. M.K. Luhandjula. Compensatory operators in fuzzy linear programming with multiple objectives. *Fuzzy Sets and Systems*, 8:245–252, 1982. 230, 234
159. T. Lyche and L.L. Schumaker. Computation of smoothing and interpolating naturel splines via local bases. *SIAM J. Numer. Anal.*, 10:1027–1038, 1973. 309
160. K. Maes, S. Saminger, and B. De Baets. On the characterization of self-dual aggregation operators. In E. Montseny and P. Sobrevilla, editors, *EUSFLAT 2005*, pages 159–164, Barcelona, Spain, 2005. European Society for Fuzzy Logic and Technology. 240, 301
161. K. C. Maes, S. Saminger, and B. De Baets. Representation and construction of self-dual aggregation operators. *Europ. J. Oper. Research*, 177:472–487, 2007. 240, 243, 301

162. J.-L. Marichal. Behavioral analysis of aggregation in multicriteria decision aid. In B. De Baets J. Fodor and P. Perny, editors, *Preferences and Decisions under Incomplete Knowledge*, pages 153–178. Physica-Verlag, Heidelberg, 2000. 240
163. J.-L. Marichal. On Choquet and Sugeno integrals as agregation functions. In M. Grabisch, T. Murofushi, and M. Sugeno, editors, *Fuzzy measures and integrals. Theory and Applications*, pages 247–272. Physica-Verlag, Heidelberg, 2000. 91, 117, 119
164. J.-L. Marichal. On Sugeno integral as an aggregation function. *Fuzzy Sets and Systems*, 114:347–365, 2000. 116
165. J.-L. Marichal. Aggregation of interacting criteria by means of the discrete Choquet integral. In T. Calvo, G. Mayor, and R. Mesiar, editors, *Aggregation Operators. New Trends and Applications*, pages 224–244. Physica-Verlag, Heidelberg, New York, 2002. 91, 93, 107
166. J.-L. Marichal. Entropy of discrete Choquet capacities. *Europ. J. Oper. Res.*, 137:612–624, 2002. 108
167. J.-L. Marichal. Tolerant or intolerant character of interacting criteria in aggregation by the Choquet integral. *Europ. J. Oper. Research*, 155:771–791, 2004. 108
168. J.-L. Marichal, R. Mesiar, and T. Rückschlossová. A complete description of comparison meaningful functions. *Aequationes Mathematicae*, 69:309–320, 2005. 302
169. R.A. Marques Pereira and R.A. Ribeiro. Aggregation with generalized mixture operators using weighting functions. *Fuzzy Sets and Systems*, 137:43–58, 2003. 67, 298
170. J. Martín, G. Mayor, and J. Torrens. On locally internal monotonic operations. *Fuzzy Sets and Systems*, 137:27–42, 2003. 202, 303
171. M. Mas, G. Mayor, and J. Torrens. T-operators. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 7:31–50, 1999. 214, 217
172. M. Mas, M. Monserrat, J. Torrens, and E. Trillas. A survey of fuzzy implication functions. *IEEE Trans. on Fuzzy Systems*, in press, 2007. 179
173. T. Masters. *Practical Neural Network Recipes in C++*. Academic Press, Boston, 1993. 314
174. G. Mayor and T. Calvo. Extended aggregation functions. In *IFSA'97*, volume 1, pages 281–285, Prague, 1997. 4
175. G. Mayor and J. Monreal. Additive generators of discrete conjunctive aggregation operations. *IEEE Trans. on Fuzzy Systems*, in press, 2007. 300
176. G. Mayor and J. Torrens. Triangular norms on discrete settings. In E.P. Klement and R. Mesiar, editors, *Logical, Algebraic, Analytic and Probabilistic Aspects of Triangular Norms*. Elsevier, New York, 2005. 300
177. D.F. McAllister and J.A. Roulier. An algorithm for computing a shape-preserving oscillatory quadratic spline. *ACM Trans. Math. Software*, 7: 331–347, 1981. 311
178. K. Menger. Statistical metrics. *Procs. Nat. Acad. Sci. U.S.A.*, 37:535–537, 1942. 126
179. R. Mesiar. Choquet-like integrals. *J. Math. Anal. Appl.*, 194:477–488, 1995. 115, 116
180. R. Mesiar. Compensatory operators based on triangular norms. In H.-J. Zimmermann, editor, *EUFIT*, pages 131–135, Aachen, 1995. 225, 227
181. R. Mesiar. Generalizations of k–order additive discrete fuzzy measures. *Fuzzy Sets and Systems*, 102:423–428, 1999. 104

182. R. Mesiar. k–order additive measures. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 7:561–568, 1999. 104
183. R. Mesiar. Fuzzy set approach to the utility, preference relations, and aggregation operators. *Europ. J. Oper. Res.*, 176:414–422, 2007. 299
184. R. Mesiar and M. Komorníková. Triangular norm-based aggregation of evidence under fuzziness. In B. Bouchon-Meunier, editor, *Aggregation and Fusion of Imperfect Information*, pages 11–35. Physica-Verlag, Heidelberg, 1998. 206, 225, 227, 230
185. R. Mesiar and T. Rückschlossová. Characterization of invariant aggregation operators. *Fuzzy Sets and Systems*, 142:63–73, 2004. 240, 301
186. R. Mesiar and D. Vivona. Two-step integral with respect to fuzzy measure. *Tatra Mount. Math. Publ.*, 16:359–368, 1999. 301
187. A. Mesiarová. *Special Classes of Triangular Norms.* PhD thesis, Slovak Technical University, 2005. 167
188. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, Berlin; New York, 2nd, extended edition, 1994. 324
189. P. Miranda and M. Grabisch. Optimization issues for fuzzy measures. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 7:545–560, 1999. 111
190. M. Mizumoto. Pictorial representations of fuzzy connectives, Part II: cases of compensatory operators and self-dual operators. *Fuzzy Sets and Systems*, 32:45–79, 1989. 240
191. B. Moser, E. Tsiporkova, and E.P. Klement. Convex combinations in terms of triangular norms. A characterization of idempotent, bysimmetrical and self-dual compensatory operators. *Fuzzy Sets and Systems*, 104:97–108, 1999. 240
192. B. Mulansky and M. Neamtu. Interpolation and approximation from convex sets. *J. Approx. Theory*, 92:82–100, 1998. 312
193. Y. Narukawa and V. Torra. Fuzzy measure and probability distributions: Distorted probabilities. *IEEE Trans. on Fuzzy Systems*, 13:617–629, 2005. 72, 103
194. Y. Narukawa and V. Torra. Graphical interpretation of the twofold integral and its generalization. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 13:415 – 424, 2005. 301
195. R. Nelsen, J.J. Quesada-Molina, J.A. Rodríguez-Lallena, and M. Ubeda-Flores. On the construction of copulas and quasi-copulas with given diagonal sections. *Mathematics and Economics*, in press, 2007. 182, 183
196. R.B. Nelsen. *An Introduction to Copulas.* Springer, Berlin, 1998. 182, 185
197. R.B. Nelsen, J.J. Quesada Molina, J.A. Rodríguez Lallena, and M. Ubeda Flores. Best-possible bounds on sets of bivariate distribution functions. *J. of Multivariate Analysis*, 90:348–358, 2004. 272
198. J. Nocedal and S. Wright. *Numerical Optimization.* Springer, New York, 1999. 317, 321
199. M.O. O'Hagan. Aggregating template or rule antecedents in real-time expert systems with fuzzy set logic. In *22nd Annual IEEE Asilomar Conf. Signals, Systems, Computers*, pages 681–689. Pacific Grove, CA, 1988. 78
200. S. Ovchinnikov and A. Dukhovny. Integral representation of invariant functionals. *J. Math. Anal. Appl.*, 244:228–232, 2000. 301
201. E. Pap, editor. *Handbook of Measure Theory.* North Holland/Elsevier, Amsterdam, Boston, 2002. 90, 91
202. P.M. Pardalos and G.C. Resende, editors. *Handbook of Applied Optimization.* Oxford University Press, New York; Oxford, 2002. 317, 321, 323, 324

203. S.A. Pijavski. An algorithm for finding the absolute extremum of a function. *USSR Comput. Math. and Math. Phys.*, 2:57–67, 1972. 318
204. J. Pinter. *Global Optimization in Action: Continuous and Lipschitz Optimization–Algorithms, Implementations, and Applications.* Kluwer, Dordrecht; Boston, 1996. 323
205. M.J.D. Powell. UOBYQA: unconstrained optimization by quadratic approximation. *Mathematical Programming*, 92:555–582, 2002. 267, 317
206. M.J.D. Powell. The NEWUOA software for unconstrained optimization without derivatives. Technical report, Technical Report DAMTP NA2004/08, Department of Applied Mathematics and Theoretical Physics, Cambridge University, Cambridge CB3 9EW, UK, 2004. 317
207. A. Pradera and E. Trillas. Aggregation, Non-Contradiction and Excluded-Middle. *Mathware & Soft Computing*, XIII:189–201, 2006. 303
208. A. Pradera and E. Trillas. Aggregation operators from the ancient NC and EM point of view. *Kybernetika*, 42:243–260, 2006. 303
209. A. Pradera, E. Trillas, and T. Calvo. A general class of triangular norm-based aggregation operators: quasi-linear T-S operators. *Intl. J. of Approximate Reasoning*, 30:57–72, 2002. 49, 230, 234
210. D. Ralescu and G. Adams. The fuzzy integral. *J. Math. Anal. Appl.*, 86: 176–193, 1980. 116
211. M. Roubens. Interaction between criteria and definition of weights in MCDA problems. In *44th Meeting of the European Working Group Multicriteria Aid for Decisions*, Brussels, Belgium, 1996. 106
212. A.M. Rubinov. *Abstract Convexity and Global Optimization.* Kluwer, Dordrecht; Boston, 2000. 324
213. W. Rudin. *Functional Analysis.* McGraw-Hill, New York, 1991. 82
214. S. Saminger. Aggregation of bi-capacities. In *10th Int. Conf. on Information Processing and Management of Uncertainty*, pages 335–342, Perugia, Italy, 2004. 299
215. S. Saminger and R. Mesiar. A general approach to decomposable bi-capacities. *Kybernetika*, 39:631–642, 2003. 299
216. N. S. Sapidis and P. D. Kaklis. An algorithm for constructing convexity and monotonicity preserving splines in tension. *Compuper Aided Geometric Design*, 5:127–137, 1988. 312
217. K. Schittkowski. *Numerical Data Fitting in Dynamical Systems: a Practical Introduction with Applications and Software.* Kluwer, Dordrecht; Boston, 2003. 321, 322
218. L.L. Schumaker. *Spline Functions: Basic Theory.* Wiley, New York, 1981. 307, 311
219. L.L. Schumaker. On shape preserving quadratic interpolation. *SIAM J. Numer. Anal.*, 20:854–864, 1983. 72, 311
220. B. Schweizer and A. Sklar. Associative functions and statistical triangle inequalities. *Publ. Math. Debrecen*, 8:169–186, 1961. 143
221. B. Schweizer and A. Sklar. *Probabilistic Metric Spaces.* North-Holland, New York, 1983. 126, 127, 143, 167
222. G. Shafer. *A Mathematical Theory of Evidence.* Princeton University press, Princeton, 1976. 102
223. B. Shubert. A sequential method seeking the global maximum of a function. *SIAM J. Numer. Anal.*, 9:379–388, 1972. 318

224. Y.-H. Shyu. *Absolute continuity in the τ-operations.* PhD thesis, Illinois Institute of Technology, 1984. 168
225. R. Sibson. A brief description of natural neighbor interpolation. In V. Barnett, editor, *Interpreting Multivariate Data*, pages 21–36. John Wiley, Chichester, 1981. 315
226. M.A. Sicilia, E. García Barriocanal, and T. Calvo. An inquiry-based method for Choquet integral-based aggregation of interface usability parameters. *Kybernetika*, 39:601–614, 2003. 105
227. W. Silvert. Symmetric summation: A class of operations on fuzzy sets. *IEEE Trans. on Systems, Man and Cybernetics*, 9:659–667, 1979. 239, 240, 241
228. A. Sklar. Fonctions de répartition à n dimensions et leurs marges. *Publ. Inst. Statist. Univ. Paris*, 8:229–231, 1959. 182
229. R. G. Strongin and Y. D. Sergeyev. *Global Optimization with Non-convex Constraints: Sequential and Parallel Algorithms.* Kluwer, Dordrecht; London, 2000. 324
230. M. Sugeno. *Theory of Fuzzy Integrals and Applications.* PhD thesis, Tokyo Inst. of Technology, 1974. 102, 162
231. T. Takagi and M. Sugeno. Fuzzy identification of systems and its application to modeling and control. *IEEE Trans. on Systems, Man and Cybernetics*, 15:116–132, 1985. 191, 193
232. V. Torra. The weighted OWA operator. *Int. J. Intelligent Systems*, 12:153–166, 1997. 71, 72
233. V. Torra. On some relationships between WOWA operator and the Choquet integral. In *8th Int. Conf. on Information Processing and Management of Uncertainty*, pages 818–824, Paris, 1998. 72
234. V. Torra. The WOWA operator and the interpolation function W*: Chen and Otto's interpolation revisited. *Fuzzy Sets and Systems*, 113:389–396, 2000. 72
235. V. Torra. Learning weights for the quasi-weighted means. *IEEE Trans. on Fuzzy Systems*, 10:653–666, 2002. 58, 59, 60, 77, 80, 87
236. V. Torra. OWA operators in data modeling and reidentification. *IEEE Trans. on Fuzzy Systems*, 12:652–660, 2004. 77
237. Y. Torra, V. Narukawa. *Modeling Decisions. Information Fusion and Aggregation Operators.* Springer, Berlin, Heidelberg, 2007. 72, 297, 301
238. J.F. Traub and H. Wozniakowski. *A General Theory of Optimal Algorithms.* Academic Press, New York, 1980. 276
239. E. Trillas. Sobre funciones de negación en la teoría de los subconjuntos difusos. *Stochastica*, III:47–59 (in Spanish) Reprinted (English version) in Advances of Fuzzy Logic, S. Barro et al. (eds), Universidad de Santiago de Compostela 31–43, 1998., 1979. 18
240. E. Trillas, C. Alsina, and A. Pradera. Searching for the roots of Non-Contradiction and Excluded-Middle. *Int. J. General Syst.*, 31:499–513, 2002. 302
241. L. Troiano and R. Yager. A meaure of dispersion for OWA operators. In Y. Liu, G. Chen, and M. Ying, editors, *Proc. of the 11th IFSA World Congress*, pages 82–87, Beijing China, 2005. Tsinghua University Press and Springer. 57, 71
242. L. Troiano and R. Yager. On some properties of mixing OWA operators with t-norms and t-conorms. In E. Montseny and P. Sobrevilla, editors, *EUSFLAT 2005*, volume II, pages 1206–1212, Barcelona, Spain, 2005. European Society for Fuzzy Logic and Technology. 249

243. L. Troiano and R. Yager. Recursive and iterative OWA operators. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 13:579–599, 2005. 70
244. L. Troiano and R. Yager. On the relationship between the quantifier threshold and OWA operators. In V. Torra, Y. Narukawa, A. Valls, and J. Domingo-Ferrer, editors, *Modeling Decisions for Artificial Intelligence*, volume LNAI 3885, pages 215–226. Springer, Heidelberg, 2006. 82
245. L. Troiano and R. Yager. On mixing the OWA operators with t-norms and t-conorms. *Draft*, 2007. 251
246. A.K. Tsadiras and K.G. Margaritis. The MYCIN certainty factor handling function as uninorm operator and its use as a threshold function in artificial neurons. *Fuzzy Sets and Systems*, 93:263–274, 1999. 200, 209
247. I.B. Turksen. Interval-valued fuzzy sets and compensatory AND. *Fuzzy Sets and Systems*, 51:295–307, 1992. 230
248. R.J. Vanderbei. *Linear Programming: Foundations and Extensions*. Kluwer, Boston, 2001. 320, 325
249. C. von Altrock, B Krause, and H.-J. Zimmermann. Advanced fuzzy logic control of a model car in extreme situations. *Fuzzy Sets and Systems*, 48: 41–52, 1992. 230
250. W. Wang, Z. Wang, and G.J. Klir. Genetic algorithms for determining fuzzy measures from data. *J. Intell. Fuzzy Systems*, 6:171–183, 1998. 113
251. Y.M. Wang and C. Parkan. A minimax disparity approach for obtaining OWA operator weights. *Inform. Sci.*, 175:20–29, 2005. 80
252. Z. Wang and G. Klir. *Fuzzy Measure Theory*. Plenum Press, New York, 1992. 90, 91, 92, 101
253. Z. Wang, K.S. Leung, and J. Wang. Determining nonnegative monotone set functions based on Sugeno's integral: an application of genetic algorithms. *Fuzzy Sets and Systems*, 112:155–164, 2000. 113
254. S. Weber. A general concepts on fuzzy connectives, negations and implications based on t–norms and t–conorms. *Fuzzy Sets and Systems*, 11:115–134, 1983. 162
255. S. Weber. Measures of fuzzy sets and measures on fuzziness. *Fuzzy Sets and Systems*, 13:114–138, 1984. 116
256. E. W. Weisstein. *CRC Concise Encyclopedia of Mathematics*. CRC Press, Boca Raton, 2002. 2
257. K. Willemans and P. Dierckx. Smoothing scattered data with a monotone powell-sabin spline surface. *Numerical Algorithms*, 12:215–232, 1996. 273
258. L. Xie and M. Grabisch. The core of bicapacities and bipolar games. *Fuzzy Sets and Systems*, 158:1000–1012, 2007. 299
259. Z.S. Xu. An overview of methods for determining OWA weights. *Int. J. Intelligent Systems*, 20:843–865, 2005. 77
260. Z.S. Xu and Q.L. Da. The ordered weighted geometric averaging operator. *Int. J. Intelligent Systems*, 17:709–716, 2002. 73
261. R. Yager. Fuzzy decision making using unequal objectives. *Fuzzy Sets and Systems*, 1:87–95, 1978. 176, 177, 226
262. R. Yager. On a general class of fuzzy connectives. *Fuzzy Sets and Systems*, 4:235–242, 1980. 156
263. R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Trans. on Systems, Man and Cybernetics*, 18:183–190, 1988. 25, 39, 68
264. R. Yager. Connectives and quantifiers in fuzzy sets. *Fuzzy Sets and Systems*, 40:39–76, 1991. 55, 81

265. R. Yager. Families of OWA operators. *Fuzzy Sets and Systems*, 59:125–148, 1993. 71
266. R. Yager. Measures of entropy and fuzziness related to aggregation operators. *Inform. Sci.*, 82:147–166, 1995. 57, 71
267. R. Yager. Quantifier guided aggregation using OWA operators. *Int. J. Intelligent Systems*, 11:49–73, 1996. 81
268. R. Yager. Fusion of ordinal information using weighted median aggregation. *Int. J. Approx. Reasoning*, 18:35–52, 1998. 120, 121
269. R. Yager. Uninorms in fuzzy systems modeling. *Fuzzy Sets and Systems*, 122:167–175, 2001. 179, 200
270. R. Yager. Noble reinforcement in disjunctive aggregation operators. *IEEE Trans. on Fuzzy Systems*, 11:754–767, 2003. 191, 192, 225, 282
271. R. Yager. Generalized OWA aggregation operators. *Fuzzy Optimization and Decision Making*, 3:93–107, 2004. 74, 115
272. R. Yager. Weighted triangular norms using generating functions. *Int. J. Intelligent Systems*, 19:217–231, 2004. 176, 178, 179, 180, 226, 229
273. R. Yager. Extending multicriteria decision making by mixing t-norms and OWA operators. *Int. J. of General Syst.*, 20:453–474, 2005. 249
274. R. Yager. Using stress functions to obtain OWA operators. *IEEE Trans. on Fuzzy Systems*, in press, 2007. 82, 83
275. R. Yager and D. Filev. *Essentials of Fuzzy Modelling and Control*. J. Wiley & Sons, New York, 1994. 8, 77, 297, 298
276. R. Yager and J. Kacprzyk, editors. *The Ordered Weighted Averaging Operators. Theory and Applications*. Kluwer, Boston, 1997. 77
277. R. Yager and V. Kreinovich. Universal approximation theorem for uninorm-based fuzzy systems modeling. *Fuzzy Sets and Systems*, 140:331–339, 2003. 200
278. R. Yager and A. Rybalov. Uninorm aggregation operators. *Fuzzy Sets and Systems*, 80:111–120, 1996. 28, 200
279. R. R. Yager and A. Rybalov. Noncommutative self-identity aggregation. *Fuzzy Sets and Systems*, 85:73–82, 1997. 303, 304
280. L. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965. 7
281. L. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. Part I. *Inform. Sci.*, 8:199–249, 1975. 300
282. L. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. Part II. *Inform. Sci.*, 8:301–357, 1975. 300
283. L. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. Part III. *Inform. Sci.*, 9:43–80, 1975. 300
284. L. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28, 1978. 8, 101
285. L. Zadeh. The role of fuzzy logic in the management of uncertainty in expert systems. In M.M. Gupta, A. Kandel, W. Bandler, and J.B. Kiszka, editors, *Approximate Reasoning in Expert Systems*. Elsevier, North-Holland, 1985. 8
286. H.-J. Zimmermann. *Fuzzy Set Theory - and its Applications*. Kluwer, Boston, 1996. 8, 32, 33, 192, 269, 270
287. H.-J. Zimmermann and P. Zysno. Latent connectives in human decision making. *Fuzzy Sets and Systems*, 4:37–51, 1980. 32, 230, 273, 274, 277

Index